Accelerating the Training of Convolutional Neural Networks for Image Segmentation with Deep Active Learning

by

Weitao Chen

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2020

© Weitao Chen 2020

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Image semantic segmentation is an important problem in computer vision. However, Training a deep neural network for semantic segmentation in supervised learning requires expensive manual labeling. Active learning (AL) addresses this problem by automatically selecting a subset of the dataset to label and iteratively improve the model. This minimizes labeling costs while maximizing performance. Yet, deep active learning for image segmentation has not been systematically studied in the literature. This thesis offers three contributions. First, we compare six different state-of-the-art querying methods, including uncertainty, Bayesian, and out-of-distribution methods, in the context of active learning for image segmentation. The comparison uses the standard dataset Cityscapes, as well as randomly generated data, and the state-of-the-art image segmentation architecture DeepLab. Our results demonstrate subtle but robust differences between the querying methods, which we analyze and explain. Second, we propose a novel way to query images by counting the number of pixels with acquisition values above a certain threshold. Our counting method outperforms the standard averaging method. Lastly, we demonstrate that the previous two findings remain consistent for both whole images and image crops.

Furthermore, we provide an in-depth discussion of deep active learning and results from supplementary experiments. First, we studied active learning in the context of image classification with the MNIST dataset. We observed an interesting phenomenon where active learning querying methods perform worse than random sampling in the early cycles but overtake random sampling at a break-even point. This break-even point can be controlled by varying model capacity, sample diversity, and temperature scaling. The difference in performances of the six querying methods is larger than in the case of image segmentation. Second, we attempt to explore the theoretical optimal query by querying samples with the lowest accuracy and querying with a trained expert model. Although they turned out to be suboptimal, their results would hopefully shed light on the subject. Lastly, we present the experiment results from using SegNet and FCN. With these architectures, our querying methods did not perform any better than random sampling. Nevertheless, those negative results demonstrate some of the difficulties of active learning for image segmentation.

Acknowledgements

I would like to thank Prof. Krzysztof Czarnecki, Dr. Sean Sedwards, Dr. Rick Salay, and Dr. Vahdat Abdelzad who made this thesis possible.

Dedication

This is dedicated to my family and friends.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation and Overview

Computer vision has many applications in areas such as autonomous driving and robotics. In the field of computer vision, semantic image segmentation is an important problem for scene understanding. Recently, deep learning has shown promising results in semantic segmentation. However, supervised deep learning requires a large amount of labeled data. Labeling images for semantic segmentation by human annotators is both expensive and time-consuming relative to data collection [7].

Active Learning (AL) [6] addresses this problem by querying the optimal subset of collected data to label. Given a set of unlabeled data, AL aims to find a small subset that gives the most accurate model [42]. The process can be iterative, where each query is based on the model learned from the previous querying cycle, hence the name active learning. Thus, the cost of labeling is minimized while the performance is maximized.

Currently, active learning for semantic segmentation with deep neural networks is not extensively studied in the literature. Most studies in this area focus on image classification [15, 25, 36, 43, 44], while the few that do focus on semantic segmentation with deep learning have limited querying methods and outdated network architectures [32, 17, 46].

This thesis contributes a comparison between multiple querying methods for their effectiveness in active learning. We also propose a novel way to query images by counting the number of pixels with acquisition values above a certain threshold. Our results are then compared between querying whole images and image crops. The predictor used is DeepLabv3+ [5], a state-of-the-art deep neural network for semantic segmentation. The

dataset is Cityscapes [7], a collection of street-view images captured from a vehicle in cities and their labels with fine annotations.

This thesis gives a brief background and related work on semantic segmentation and active learning. Following that, we describe the methods used in this research including each acquisition function, network architecture, and training process. Thereafter, the experimental results are presented and discussed. Additionally, we provide an in-depth discussion of deep active learning and results from supplementary experiments. First, we studied active learning in the context of image classification with the MNIST dataset. Then, we attempt to explore the theoretically optimal querying. Lastly, we present results from experiments with other network architectures.

1.2 Background and Related Work

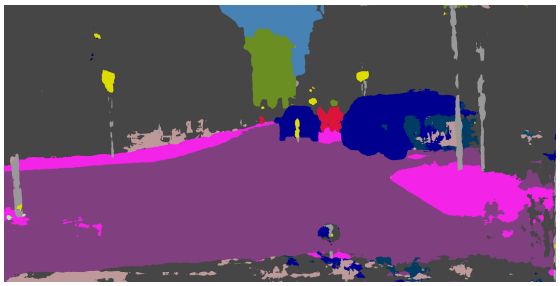
1.2.1 Semantic Image Segmentation

Semantic image segmentation [12] is the process of classifying each pixel in a given image. It is a special case of classification problem as each input image has multiple output classes for each pixel. It is also a more difficult problem than image classification because not only do we need to know what is in a given image, we also need to know where the objects are and their precise contours. The set of classes is predefined and each pixel must be labeled as one of the classes. An exception is the "ignore" class. Pixels labeled as "ignore" class in the ground truth will not affect evaluation.

Traditional methods involve clustering groups of neighboring pixels together into superpixels [41]. For more recent state-of-the-art methods, deep learning is at the heart of image segmentation. Training deep neural networks by learning from examples through the process of supervised learning yields unprecedented accuracy. During training, a pool of labeled images called the training set is used to tune the parameters of the neural network through the process of backpropagation [20]. The number of parameters in a neural network is correlated to its model capacity or degrees of freedom. Model capacity is a measure of how well it can model complicated relationships. However, it is not directly proportional to the number of parameters since other factors such as the exact connections need to be considered.

After the weights have converged, we use a separate pool of labeled images for evaluation. The labels annotated by humans and used for training are called ground truth. The labels assigned by the neural network are called predictions. The reason we use a separate

(a) Input image (b) Ground truth



(c) Prediction

Figure 1.1: Example image segmentation results

data pool is that we want to generalize our ability to accurately predict new images instead of overfitting on the training set. Usually, a test set is used for evaluation and a validation set is used for tuning hyperparameters. However, in our case, we use the validation set as the test set, because the ground truth of Cityscapes' test set is not available. Figure 1.1 gives an example of an input image, ground truth, and output prediction from our deep neural network.

A common metric for evaluating semantic segmentation, which is adopted in this thesis, is the mean Intersection-Over-Union (mIOU) [1]. mIOU is the mean of IOU of each class, and IOU is defined as follows: $IOU = \#true\_positive/(\#true\_positive + \#false\_positive + \#false\_negative)$. The reason we prefer mIOU over simple percentage pixel accuracy is that smaller objects could be equally important as larger ones. For example, if the prediction of the neural network completely ignores small objects and only labels large objects correctly, it will have an unreasonably high percentage pixel accuracy, but the mIOU score will be heavily penalized. In the case of autonomous driving, small objects like lampposts or small children should be equally important as a large truck.

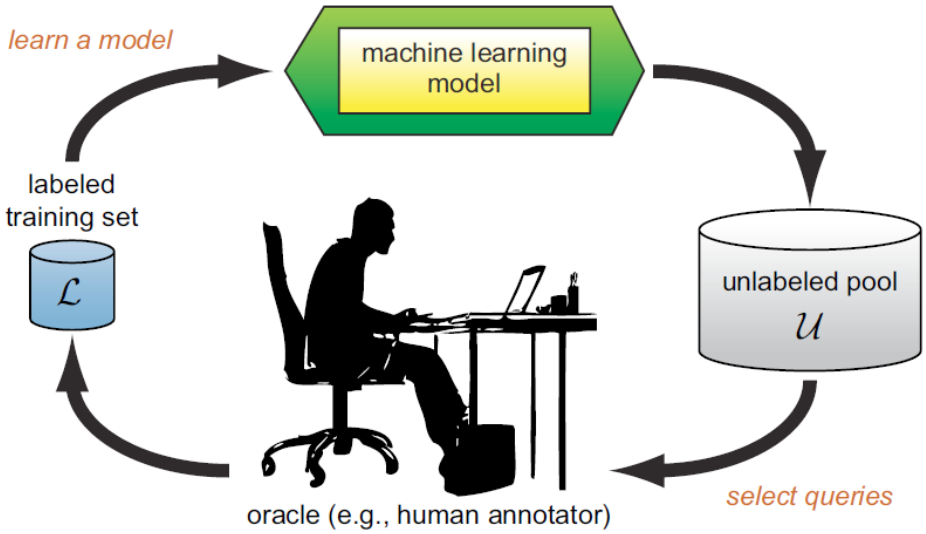


Figure 1.2: Pool-based active learning cycle
Source: Adapted from [37]

1.2.2 Active Learning

Active learning is described in detail in the survey by Settles [37]. In pool-based active learning (figure 1.2) [28], which is our focus, a pool of data is collected but not labeled. Active learning queries the unlabeled pool to select samples to be labeled by an oracle, who is usually a human annotator. The newly labeled data are then added to the labeled training set. Using the training set, a machine learning model is trained through supervised learning. The model learned is then evaluated on the validation set. If the performance on the validation set achieved is satisfactory or if the budget for labeling is spent, the process is stopped. If not, the model is used to query new samples from the unlabeled pool and the whole cycle starts again.

There are many different ways to query. The most trivial method is random querying. To perform better than the random baseline, a querying method could examine each datum and assign a value that estimates its information gain with an acquisition function. Since one wants to maximize the information gain, usually the top data with the largest acquisition values are chosen. In some cases, however, it might be beneficial to query part of data with lower acquisition values. The mixture of querying high and low acquisition values is called sample diversity. Fu et al. have a good example of the effects of sample

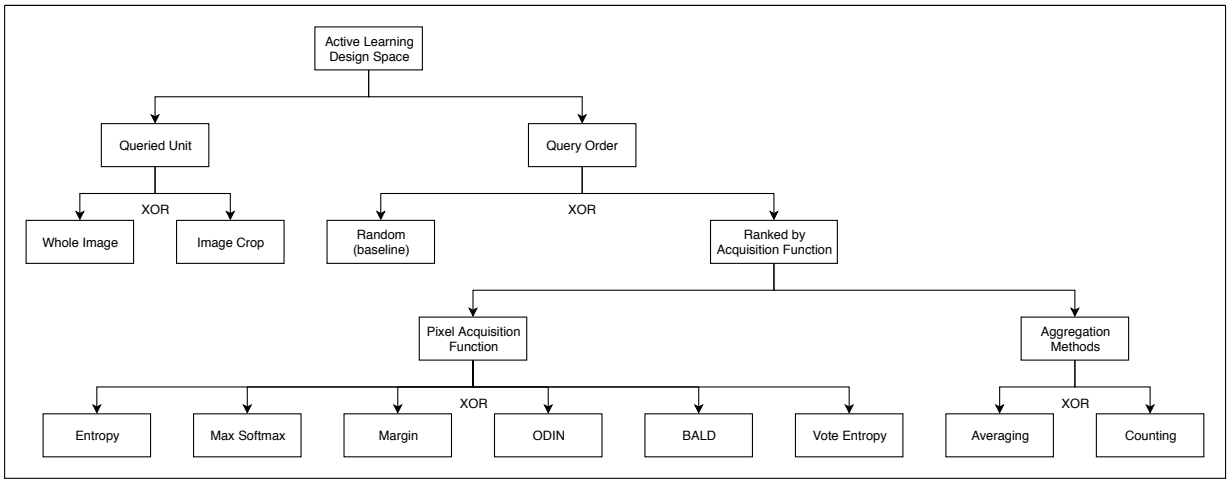


Figure 1.3: Experiment design tree

diversity in their paper [13].

1.2.3 Active Learning for Semantic Segmentation

Active learning for image segmentation has similar ideas as active learning for classification. The only difference is that each image has multiple pixel predictions, the labeling cost for each image could be different, and crops of images could be queried in region-based active learning [32, 23]. In this thesis, however, we assume the labeling cost for each image is the same and images must be cropped before the first active learning cycle begins.

Active learning for semantic segmentation with deep learning is explored in [17, 46, 32]. The first two approaches [17, 46] have only foreground and background segmentation, while the third approach is applied to Cityscapes. Marc Gorriz et al. [17] used U-net [33] for the network and Monte-Carlo dropout for querying. Lin Yang et al. [46] used Fully Convolutional Network (FCN) [31] for network and bootstrapping [11] as uncertainty measure with cosine similarity for sample diversity. Mackowiak et al. [32] also used FCN for the network and explored entropy querying and vote entropy querying.

1.3 Research Questions and Methodology

Our research objective is to examine and compare different active learning designs in the context of image segmentation. More specifically, it is to answer the question of what is the best way to label a pool of data to train a deep learning model to achieve the highest performance with the lowest labeling cost. To answer this question, we perform experiments that explore the AL design space, which consists of three main dimensions: queried unit type, pixel acquisition function, and aggregation method. Each dimension has multiple choices, and thus the best active learning design is a combination of choices from each design dimension.

Figure 1.3 illustrates our experiment design tree. For each active learning experiment, a querying unit can be either a batch of whole images or image crops. In this thesis, the word "image" refers to both whole image and image crop, unless stated otherwise. The querying order can be either random, our baseline, or ranked by an image acquisition function. Each image acquisition value is aggregated from its pixel acquisition values either by averaging or by counting values over a threshold. We explore 6 different pixel acquisition functions in addition to random selection. Overall, the diagram represents 26 different querying approaches.

1.3.1 Querying Methods

The querying methods compared are random, entropy [38], max-softmax [21], margin [35], ODIN [29], BALD [22], and vote entropy [8]. These methods are chosen because they are the state-of-the-art active learning querying method, uncertainty measure, or Out-Of-Distribution (OOD) measure. Additionally, the methods have to be computationally feasible for semantic segmentation, and they must not alter the network architecture in any way that affects its performance.

Each subsection will describe the querying method briefly. All of the acquisition functions assign an acquisition value to a single pixel. These pixel acquisition values are then aggregated together. The aggregation methods are described in the next section.

Random

Random querying is used as the baseline for all comparisons. Each image in the unlabeled pool has an equal probability to be queried. The acquisition function is a random number generator of uniform distribution from 0 to 1.

Entropy

Entropy [38] is the most common uncertainty measure used for active learning. The closer the distribution is to uniform, the more uncertain the model is, and the higher entropy it has. The per-pixel acquisition function for entropy querying is:

$$S = -\sum_{c} P_c \log P_c \tag{1.1}$$

The probability P is the softmax output of neural network, and c is the class of label.

Max-softmax

Max-softmax is an OOD measure in image classification [21]. Only the probability of the most likely class is considered. It is also an uncertainty measure on the model's confidence in labeling the data correctly. By querying the highest max-softmax score, we pick the samples that are the least confident. The per-pixel acquisition function for max-softmax querying is:

$$S = 1 - \max_{c}(P_c) \tag{1.2}$$

Margin

Margin sampling [35] takes the difference between the top two class probabilities. It measures the ambiguity between the top two classes. The per-pixel acquisition function for margin querying is:

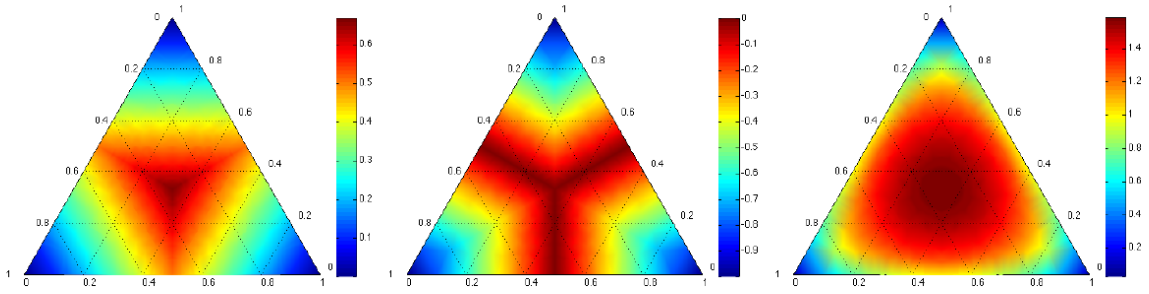
$$S = 1 - (P_1 - P_2) \tag{1.3}$$

where P_1 and P_2 are the first and second highest class probability.

Entropy, max-softmax, and margin are related to each other. Figure 1.4 shows a visualization of the uncertainty measure on simplex for a three-label classification problem [37].

ODIN

ODIN (Out-of-Distribution detector for Neural networks) [29] is an improvement upon max-softmax. The procedure is similar to max-softmax, but the softmax function is scaled



(a) least confident (b) margin (c) entropy

Figure 1.4: Uncertainty measure simplex
Source: Adapted from [37]

by temperature T. T is hyperparameter that could be calibrated for a neural network model to give a more accurate confidence prediction. [18] The new softmax function is:

$$P_i(\boldsymbol{x}; T) = \frac{\exp\left(f_i(\boldsymbol{x})/T\right)}{\sum_{j=1}^{N} \exp\left(f_j(\boldsymbol{x})/T\right)} \tag{1.4}$$

$f_c(x)$ is the output of the neural network before softmax for class c. The input image x is also preprocessed by adding a small perturbation scaled by ε.

$$\tilde{\boldsymbol{x}} = \boldsymbol{x} - \varepsilon \text{sign}(-\nabla_{\boldsymbol{x}} \log P_{\hat{y}}(\boldsymbol{x}; T)) \tag{1.5}$$

$$P_{\hat{y}}(\tilde{\boldsymbol{x}}; T) = \max_i P_i(\tilde{\boldsymbol{x}}; T) \tag{1.6}$$

The per-pixel acquisition function for the ODIN querying is the same as max-softmax querying.

BALD

BALD (Bayesian Active Learning by Disagreement) [22] is a Bayesian active learning technique for image classification [15]. Bayesian network utilizes dropout [40], which is normally a stochastic regularization technique used during training. In BALD, dropout is performed during training as well as inference with Monte-Carlo (MC) dropout [14]. The

result is a committee of different models from the same network. The per-pixel acquisition function for BALD querying is:

$$\mathbb{I}[y, \boldsymbol{\omega}|\mathbf{x}, \mathcal{D}_{\text{train}}] = \mathbb{H}[y|\mathbf{x}, \mathcal{D}_{\text{train}}] - \mathbb{E}_{P(\boldsymbol{\omega}|\mathcal{D}_{\text{train}})}\big[\mathbb{H}[y|\mathbf{x}, \boldsymbol{\omega}]\big] \tag{1.7}$$

$\mathbb{H}[y|\mathbf{x}, \mathcal{D}_{\text{train}}]$ is the entropy of y given model weights $\boldsymbol{\omega}$, input $\mathbf{x}$, and pool data $\mathcal{D}_{\text{train}}$.

Vote Entropy

Vote Entropy [8] applies Bayesian network and MC dropout similar to BALD. The per-pixel acquisition function for vote entropy querying is:

$$V^{(u,v)} := -\sum_c \frac{\sum_e D(P_e^{(u,v)}, c)}{N_E} \cdot log \frac{\sum_e D(P_e^{(u,v)}, c)}{N_E} \quad where \quad D(a,c) = \begin{cases} 1, & \text{if } argmax(a) = c \\ 0, & \text{otherwise} \end{cases} \tag{1.8}$$

Variable e is an instance of committee, N_E is the number of models in the committee. $P_e^{(u,v)}$ is the softmax output of instance e of the neural network on pixel (u, v).

1.3.2 Aggregation Methods

After the acquisition value for each pixel is evaluated, all pixel acquisition values in an image need to be aggregated into a single image acquisition value. Choosing a good aggregation method can be difficult because it is impossible to know the exact importance of each pixel before labeling. Thus, a simple and reasonable assumption is that each pixel has equal importance, and the image acquisition value is simply the average of all pixel acquisition values.

The image acquisition value of aggregation by averaging is given by

$$S_{image} = \sum S_{pixel}/N_{pixel} \tag{1.9}$$

where S_{pixel} is the pixel acquisition value, and N_{pixel} the number of pixels in the given image.

We propose a novel aggregation method by counting. Instead of taking the average, we set a threshold of a and count the number of pixels that have a pixel acquisition value higher than a. This gives an estimate of the area of informative regions. The image acquisition value of aggregation by counting is given by

$$S_{image} = \sum D(S_{pixel}, a)/N_{pixel} \quad where \quad D(s,a) = \begin{cases} 1, & \text{if } s > a \\ 0, & \text{otherwise} \end{cases} \tag{1.10}$$

1.3.3 Querying Unit

Mackowiak et al. showed that querying image crops outperforms querying whole images with the same total amount of pixels [32]. Although we did not replicate their entire experiment with region-based active learning, we experimented with different querying methods and aggregation methods on a pool of random crops. For every whole image in the dataset, we randomly cropped a single 512×512 crop, 8 times smaller than the whole image, and discarded the rest of the image. The collection of random crops are now treated as the new data pool. During each active learning cycle, each crop is treated as a whole image but with a different resolution, and everything else is kept the same for comparison. The evaluations are still performed on the original validation set with whole images.

The difference between our approach and region-based active learning is that in region-based AL, a sliding window is used to scan all the whole images. Crops within the sliding window are selected deliberately by a metric, and every selected crop is labeled. Thus, cropping becomes part of the image querying process, whereas in our approach, the image querying process comes after cropping.

Chapter 2

Experimental Comparison of Active Learning Query Methods

In this chapter, we attempt to answer our research question: what is the best active learning query method in the context of image segmentation? We will explain our experiment setup including the network architecture and the training process. Then, we will present our results from exploring our experiment design space.

2.1 Experiment Setup

2.1.1 Network Architecture

The network architecture used in this thesis is DeepLabv3+. DeepLabv3+ extends DeepLabv3 with encoder-decoder structure [5]. It employs techniques from previous versions including atrous convolution and atrous spatial pyramid pooling (ASPP) [2, 3, 4]. Figure 2.1 shows the network architecture. Currently, DeepLabv3+ has state-of-the-art results on semantic image segmentation datasets including Cityscapes [5].

The backbone architecture used is resnet_v1_50_beta, which modifies ResNet-101 [19] by replacing the first 7×7 convolution with three 3×3 convolutions [5]. The weights are pretrained on ImageNet [34].

During training, the batch size used is 8, the base learning rate is 0.007, weight decay is 0.0001, atrous rates are 6, 12, and 18, output stride is 16, and decoder output stride is 4.

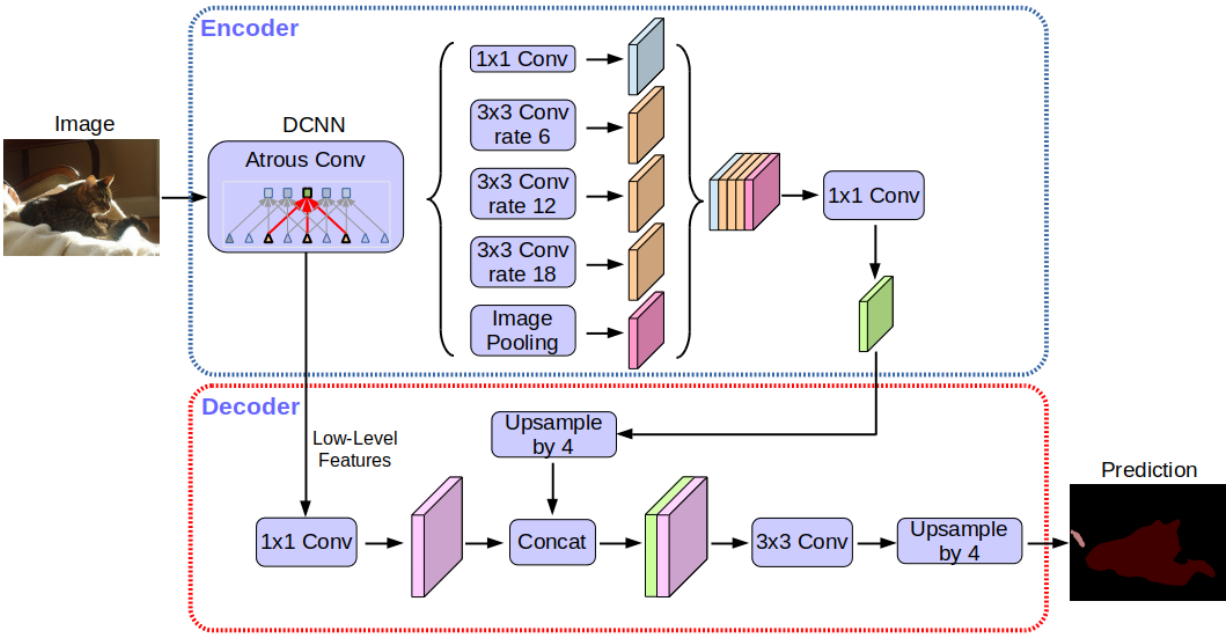


Figure 2.1: DeepLabv3+ architecture
Source: Adapted from [5]

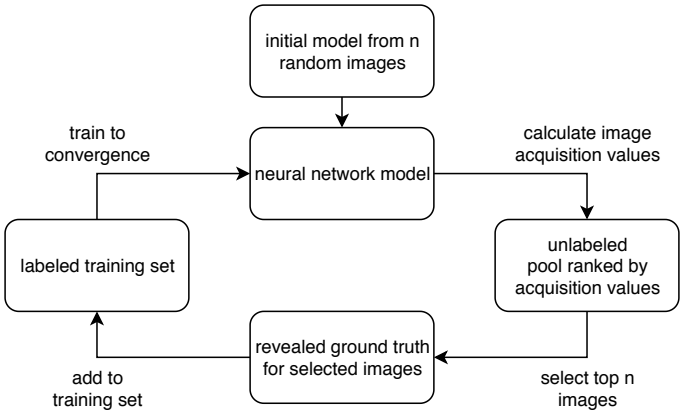


Figure 2.2: Active learning cycle

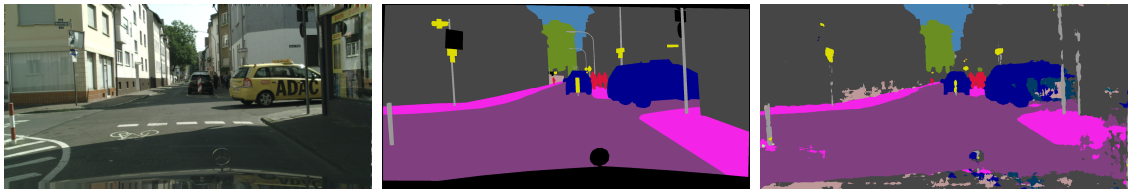
For data preprocessing, every image is randomly cropped to 513×513, and randomly flipped and scaled from 0.5 to 2. For this setup, a model fully-trained on the entire Cityscapes training set can achieve around 75% mIOU.

For BALD, the dropout rate is 0.9 for training, and 0.5 for inference. The difference in dropout rates between training and inference are compensated by weight scaling. The number of models in the committee is 10 for both BALD and vote entropy.

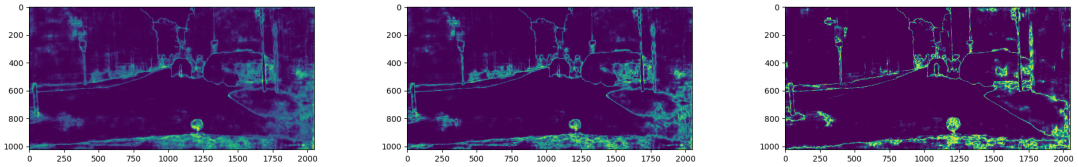
2.1.2 Training Procedure

Since the Cityscapes dataset is already fully labeled, the ground truth is simply hidden until declared labeled. This way, there is no need for human annotator and the oracle is part of the program. The dataset is divided into 2975 in the training set and 500 in the validation set.

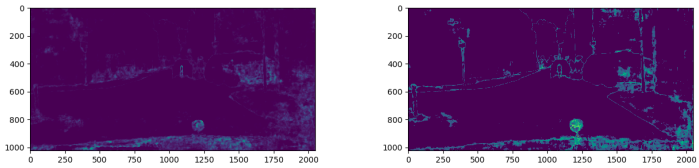
Since training for image segmentation is computationally expensive, querying images one by one is not feasible. Therefore, a batch of size $n = 50$ is queried in every active learning cycle. This is not to be confused with the training batch size. For image crops, since each crop is 8 times smaller than a whole image, so the querying batch size is 8 times larger ($n = 400$). This ensures that each query acquires the same amount of pixels. Every experiment starts with the same model trained to convergence with n randomly selected images. This ensures all querying methods have a fair comparison. The initial model is selected from 5 sets of 50 randomly selected images, and each set is trained two times. Out of the 10 runs, the performance ranged from 50% mIOU to 52% mIOU, and the one



(a) Input image (b) Ground truth (c) Prediction



(d) Entropy map (e) Max-softmax map (f) Margin map

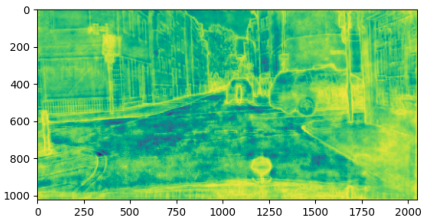


(g) BALD map (h) Vote entropy map

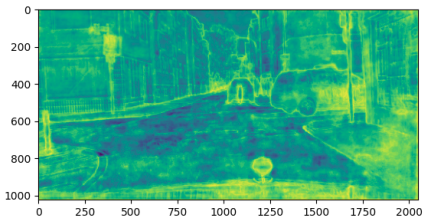
Figure 2.3: Example output images with DeepLab

with the average performance of around 51% mIOU is picked to be the initial model for all experiments.

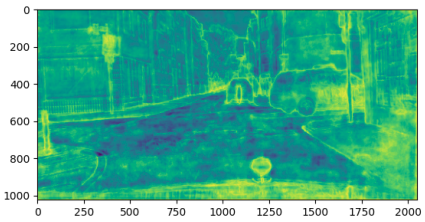
Figure 2.2 shows the typical active learning cycle. In the first cycle, the common initial model is used to query the next n images with the highest acquisition value. These selected images have their ground truth revealed and added to the labeled training set. Now with the combined training set, the neural network is trained again from scratch to convergence. The cycle then repeats until some stopping criteria are met.



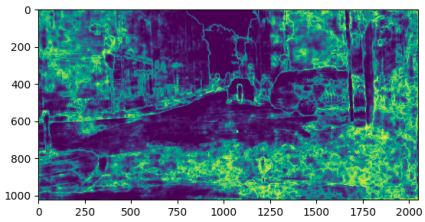
(a) $T = 10$, $epsilon = 0$



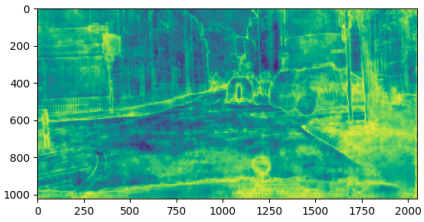
(b) $T = 100$, $epsilon = 0$



(c) $T = 1000$, $epsilon = 0$



(d) $T = 1$, $epsilon = 0.0014$



(e) $T = 1000$, $epsilon = 0.0014$

Figure 2.4: ODIN acquisition value map

2.2 Results

The experiments are divided into three parts, each corresponding to a dimension of our experiment design. The first part examines all of the pixel acquisition functions with aggregation by averaging on whole images. We compare the three uncertainty measures: entropy, max-softmax, and margin querying, then the two Bayesian methods: BALD and vote entropy. ODIN is studied separately to demonstrate the effect of temperature and epsilon. For the second part, the most promising methods from the first part are selected to apply aggregation by counting. In the final part, aggregation by counting is compared with averaging on image crops. All experiments are compared with the random baseline as reference. Figure 2.3 and 2.4 shows an example input image and ground truth, along with the model's prediction and pixel acquisition value maps of different acquisition functions.

To evaluate a given querying method, we can look at the learning curve, which plots the model's mIOU on the validation set as a function of the number of labeled data in the training set for each active learning cycle. Since we want to save labeling costs, the best querying method would have a learning curve that rises the quickest. Eventually, all querying methods will converge to the performance of a model trained on the entire data pool (75%). All of our learning curves are averaged over 4 different runs. The variance between each run is insignificant. Another evaluation metrics is the area under the learning curve (ALC). Since the fastest growing learning curve has the highest ALC, a higher ALC indicates a better querying method when the learning curves do not intersect. The advantage of using ALC is that it summarizes the learning curve into a single number.

From our results, we want to answer three questions. First, does using uncertainty and OOD measures for querying methods improve the performance of active learning over the random baseline? And how do they compare among themselves? Second, how does our proposed image acquisition value aggregation method by counting compare to the existing aggregation by averaging? Lastly, do the answers for the previous questions hold regardless of whether the queried units are whole images or image crops?

2.2.1 Aggregation by Averaging on Whole Images

In the first experiment, all of the pixel acquisition functions with aggregation by averaging on whole images are compared. Figure 2.5 summarizes the learning curves of all querying methods experimented in this thesis. Figure 2.6 normalizes these learning curves by scaling each point as a percentage of range from the first cycle mIOU to the maximum mIOU. Section 2.2.3 explains why the maximum mIoU for whole images vs. crops differ. Since

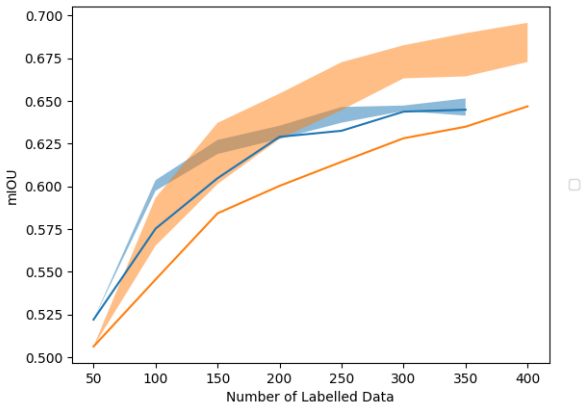


Figure 2.5: Learning curves

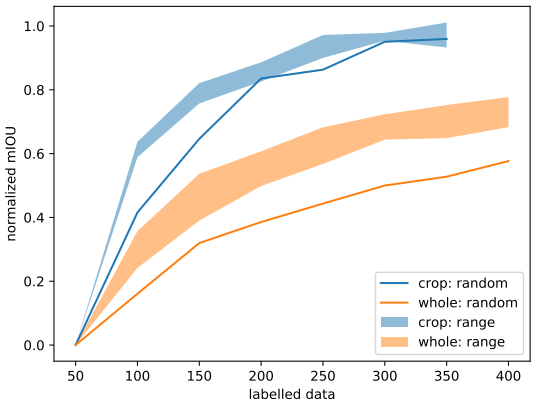


Figure 2.6: Normalized learning curves

Querying Method	ALC
Random	238.0
Entropy	255.0
Entropy: $a > log(2)$	256.5
Margin	253.0
Margin: $a > 0.8$	254.6
Margin: $a > 0.9$	254.1
Max Softmax	253.8
Max Softmax: $a > 0.5$	254.9
ODIN: $T = 0.1$	253.4
ODIN: $T = 10$	249.5
ODIN: $T = 100$	251.7
ODIN: $T = 0.01$	252.7
ODIN: $T = 1000$	252.0
ODIN: $T = 1000, epsilon = 0.0014$	247.9
BALD	255.5
Crop Random	212.6
Crop Entropy	215.6
Crop Entropy: $a > log(2)$	215.6
Crop Margin	215.2
Crop Margin: $a > 0.8$	215.6
Crop Max Softmax	215.2
Crop Max Softmax: $a > 0.5$	215.8
Crop ODIN: $T = 0.1$	215.8

Table 2.1: ALC

Querying Method	ALC
Random	145.7
Entropy	215.4
Entropy: $a > log(2)$	221.7
Margin	207.3
Margin: $a > 0.8$	213.7
Margin: $a > 0.9$	211.6
Max Softmax	210.6
Max Softmax: $a > 0.5$	214.9
ODIN: $T = 0.1$	208.9
ODIN: $T = 10$	192.8
ODIN: $T = 100$	201.9
ODIN: $T = 0.01$	205.9
ODIN: $T = 1000$	203.1
ODIN: $T = 1000 epsilon = 0.0014$	186.4
BALD	217.7
Crop Random	233.5
Crop Entropy	256.8
Crop Entropy: $a > log(2)$	257.2
Crop Margin	253.9
Crop Margin: $a > 0.8$	257.1
Crop Max Softmax	254.2
Crop Max Softmax: $a > 0.5$	258.6
Crop ODIN: $T = 0.1$	258.8

Table 2.2: Normalized ALC

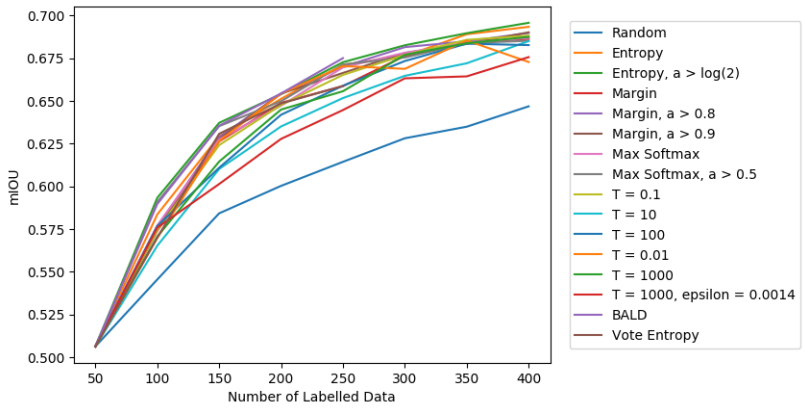


Figure 2.7: Learning curves of whole image querying unit

many learning curves are clumped close together, we plot the range of the querying methods in a transparent band for easier viewing. The random baselines are shown in solid lines. The orange curves query whole images, and the blue curves query image crops.

Immediately, we can see that all pixel acquisition functions perform better than the random baseline at every active learning cycle. There is a significant gap between the band of querying methods and the solid baseline. This shows that using any querying methods in this thesis, we can get a higher mIOU than using random querying at any amount of labeled data. For example, at 250 whole images, entropy querying outperforms random querying by 6% mIOU. From a different perspective, using any querying methods can also save the labeling cost at any mIOU level. For example, at 63% mIOU, entropy querying used 50% less labeled data compared to random querying.

However, the differences among the querying methods themselves are quite small. We can see that from the narrowness of the band of learning curve range. Most of the learning curves are concentrated at the top layer of the band. Figure 2.7 plots the individual learning curves of the querying methods with the random baseline. The fact that it is hard to distinguish one curve from another shows us that the differences are small.

Furthermore, Table 2.1 lists the ALC of all querying methods experimented within this thesis. Table 2.2 is the normalized version of table 2.1. The querying methods with crop in front of their names query image crops, while the rest query whole images. The querying methods that specify threshold a use aggregation by counting, while the rest use aggregation by averaging. Again, we see that the ALCs of the querying methods with aggregation by average are quite close to each other, ranging from 247.9 to 255.5. On the other hand, random querying is much lower at 238.0. The ALC confirms with our

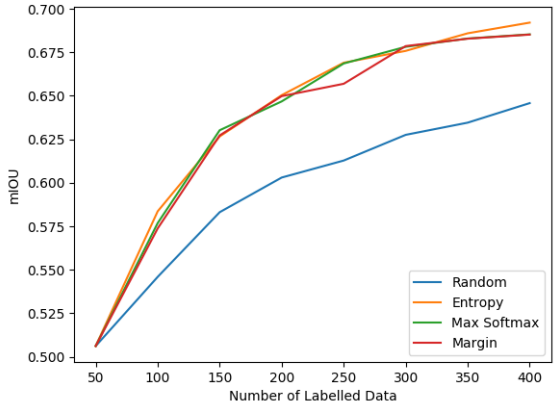


Figure 2.8: Learning curves of uncertainty querying methods

previous observation that all querying methods are better than random querying, but the differences among themselves are small.

The natural question to ask is why are the performances of different querying methods so similar. After all, they have different acquisition functions, each with different properties and purposes ranging from uncertainty measure to OOD to Bayesian inference. One plausible explanation is that we are reaching the limit of active learning given the querying units are whole images. Given any data pool, there must exist at least one querying order that gives us the optimal ALC. Chances are that optimal ALC is not much higher than the highest ALC in our querying methods. Having said that, there are cases where small improvements in mIOU matter. Also, combined with other dimensions of active learning such as region-based AL, having a better querying method could make a bigger difference.

Next, we look at different groups of querying methods in more detail. The three uncertainty measures are entropy, max-softmax, and margin. Figure 2.8 plots the learning curves of these three querying methods with the random baseline. The three curves are indistinguishable from one another. This could be caused by these three uncertainty measures being closely related. On the extreme ends, a uniform distribution has the maximum pixel acquisition values in all three measures, while a single peak distribution has the minimum (Figure 1.4). This similarity in the behaviors on the extremes may be the cause of the similarity in the ultimate performances in active learning.

The two Bayesian methods vote entropy and BALD show statistically significant, albeit small, differences. Figure 2.9 plots the learning curves of these two querying methods with the random baseline. We also put entropy querying as a reference. Vote entropy

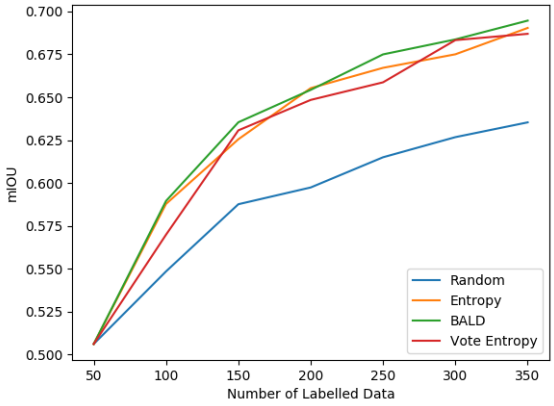


Figure 2.9: Learning curves of Bayesian querying methods

querying performed no better than entropy querying, which is in agreement with the result from Mackowiak et al. [32]. However, BALD performed better than vote entropy and entropy in every active learning cycle. This is likely because BALD has relatively low acquisition values for the pixels around the contour of objects or the boundary where one class transitions into the next. These contour pixels have high aleatoric uncertainty [10], but repeatedly training these pixels may not be beneficial. This makes BALD one of the best querying methods by a small margin.

ODIN querying is a more interesting case. Liang et al. [29] showed that as T increases, OOD detection performance increased monotonically although the effect diminishes when T is too large [29]. We observed that this is incorrect in the context of active learning based on our experiments. We selected 3 different T and repeated the experiment 4 times. Figure 2.10 plots the averaged learning curves of $T = 0.1, 10, 100$ along with the random baseline and max-softmax querying. Since max-softmax is equivalent to $T = 1$, the order of performance of T from best to worst is 1, 0.1, 100, 10. This shows that the relationship is not monotonic.

To demonstrate a larger range of T and the effect of epsilon perturbation, we ran more experiments but did not repeat them. Figure 2.11 plots the learning curves of ODIN querying with different temperature T value and epsilon perturbation. We also put the random baseline and max-softmax querying for reference. Max-softmax querying is equivalent to ODIN querying with $T = 1$ and $epsilon = 0$. Liang et al. showed that a large T and a moderate epsilon is the best for OOD detection [29]. We first tried $T = 1000$ and $epsilon = 0.0014$ which worked well for their dataset. For active learning, however, these

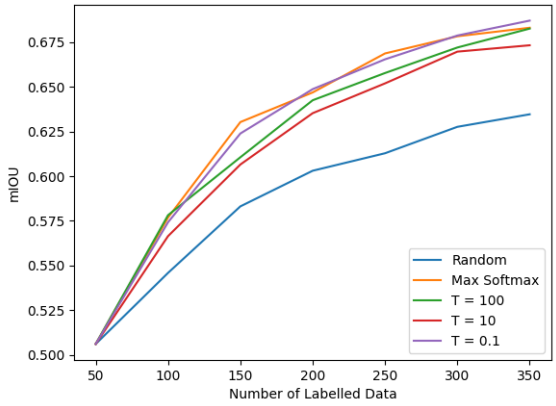


Figure 2.10: Average learning curves of ODIN queries

parameters did significantly worse than max-softmax. Again, this shows the results from Liang et al. do not transfer to active learning.

To generalize the effect of temperature scaling to other datasets, we randomly generated a pool of softmax distributions skewed towards high entropy. Each softmax distribution is treated as a querying unit, and the pixel acquisition functions can be directly applied to it. Figure 2.12 plots the normalized cumulative average of various acquisition values against entropy. For a single image, the x-axis represents the entropy of individual pixel. By ordering the pixels in increasing entropy, we can take the average of acquisition values of pixels encountered so far. The normalized cumulative average curve of max-softmax is quite close to the entropy curve, and this is reflected in their similar performance. $T = 10$ is the furthest away from the entropy curve on one side, even further than $T = 100$. This again matches our previous finding that $T = 10$ had the worst performance. The fact that a randomly generated dataset can predict the performance of active learning on real datasets tells us that our results are not dataset nor network dependent.

2.2.2 Aggregation by Counting on Whole Images

For the second part of our experiment, we selected the three uncertainty acquisition functions for aggregation by counting. For entropy querying, we experimented with a threshold $a > log(2)$. The entropy at this particular threshold corresponds to a distribution with two classes with an equal probability of 0.5, and all other classes being 0. We get $-2 * 0.5 * log(0.5) = log(2)$. This is an important threshold because a lower entropy is more

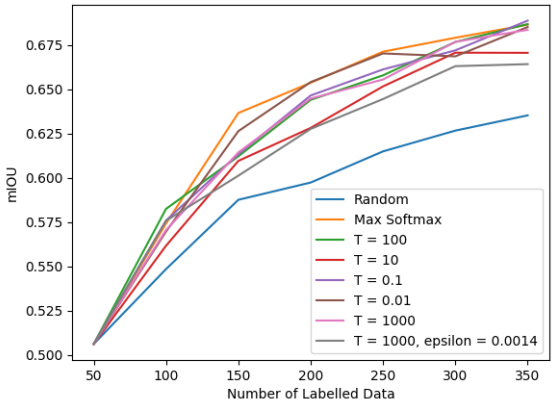


Figure 2.11: Learning curves of ODIN queries

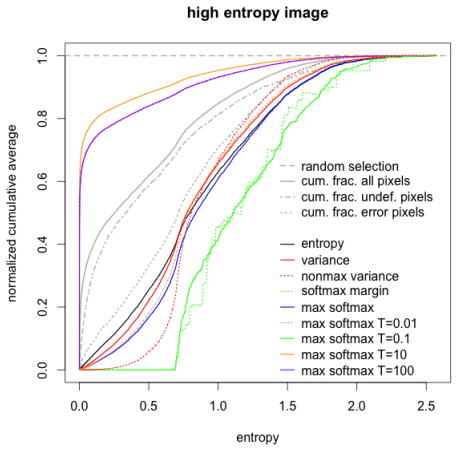


Figure 2.12: Normalized cumulative average of acquisition values

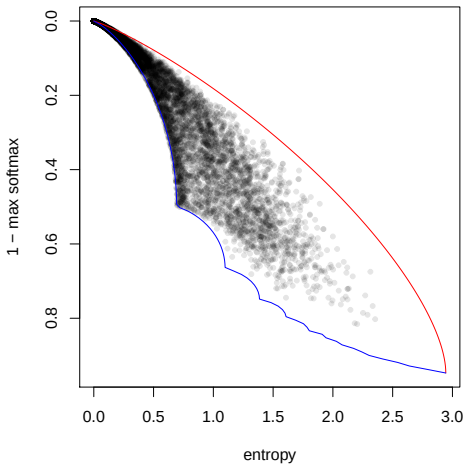


Figure 2.13: Scatter plot of (1 - maxsoftmax) vs. entropy

likely to have a single distinctive peak in the class probability distribution. Entropy querying with aggregation by counting performed better than entropy querying with aggregation by average in most cycles.

In figure 2.13, we plot max-softmax against entropy for the pixels of an example image with low average entropy during early cycles of active learning. The red and blue lines indicate the boundary of maximum and minimum possible values for distributions with 19 classes. From the plot, we can see that most pixels have low entropy and high softmax. Along the lower boundary, many pixels are concentrated along the first bend. These pixels correspond to having a single high probability class and a weaker one while the rest are close to 0. By choosing $a > log(2)$, we exclude these pixels.

Figure 2.14 plots the learning curves of entropy querying by counting with $a > log(2)$ compared to the original entropy querying by averaging and the random baseline. Entropy querying with aggregation by counting performed better than entropy querying with aggregation by average in most cycles.

Next, we compare the performance of max-softmax querying. Figure 2.15 plots the learning curves of max-softmax querying by counting with $a > 0.5$ along with the original max-softmax by averaging and the random baseline. Note that the threshold is set upon the pixel acquisition value, so $a > 0.6$ means we exclude distributions with the max probability greater than 0.4. Interestingly, the distribution with two 0.5 class probability having $log(2)$ entropy also has 0.5 max-softmax. However, the two query thresholds are different. In figure 2.13, the cutoff line for entropy is vertical, whereas max-softmax is horizontal. The

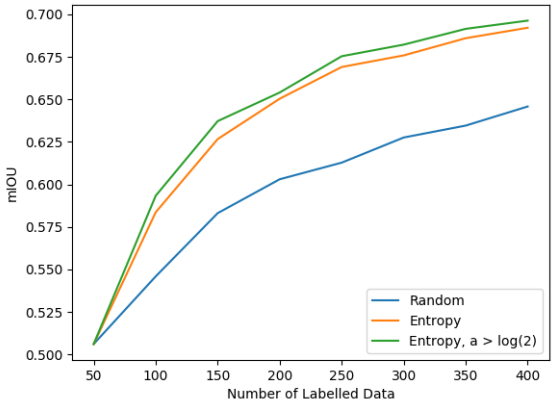


Figure 2.14: Learning curve of entropy querying with aggregation by counting

results show that max-softmax querying by counting with $a > 0.5$ has better performance than aggregation by average.

Lastly, we compare the performance of margin querying. Figure 2.16 plots the learning curves of margin querying by counting with $a > 0.8$ along with the original margin by averaging and the random baseline. Similar to max-softmax, margin with $a > 0.8$ means we exclude distributions with the difference between the top two probabilities greater than 0.2. The results show that margin querying by counting with $a > 0.8$ performed better than aggregation by averaging.

These results show that aggregation by counting perform better than the standard aggregation by averaging for entropy, max-softmax, and margin querying. Although aggregation by averaging is the most straightforward approach, problems can arise when the concentration of high pixel acquisition values is uneven. For example, consider two images A and B. Image A has a small area of high pixel acquisition values, whereas image B has a large area of medium pixel acquisition values. Since a larger image area is more likely to contain varying objects and thus more total information, it might be more beneficial to query image B instead of image A even if image A has higher average pixel acquisition value than image B. Following this logic, we can count the number of pixels with acquisition values higher than a certain threshold, and query the images with the highest pixel count. Using this method, it is important to pick the right threshold. Since different pixel acquisition functions have different range and distributions, the threshold needs to be tuned for individual pixel acquisition functions.

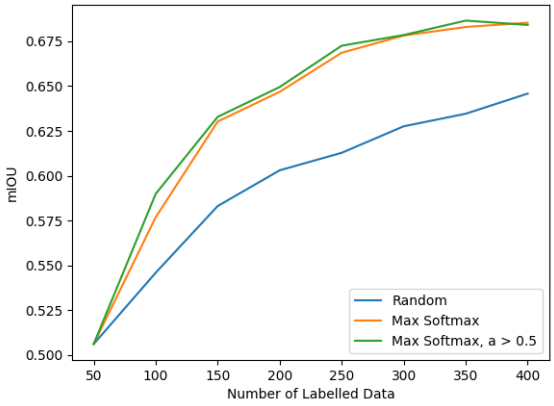


Figure 2.15: Learning curve of max-softmax querying with aggregation by counting

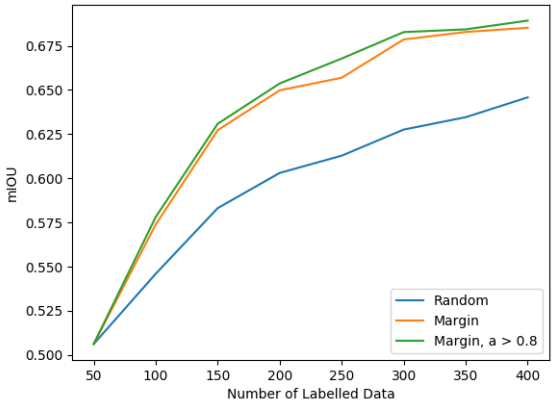


Figure 2.16: Learning curve of margin querying with aggregation by counting

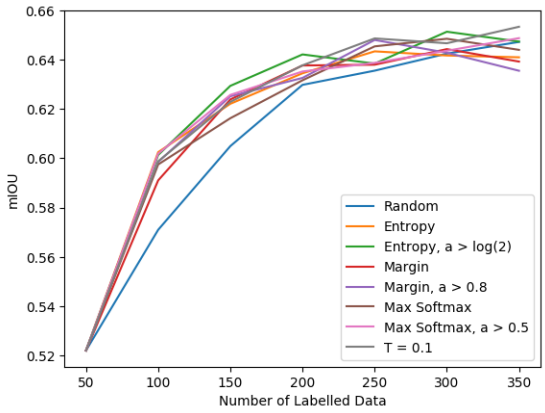


Figure 2.17: Learning curves of image crop querying unit

2.2.3 Image Crop

For the final part of our experiment, we compare both aggregation methods on image crops. Figure 2.17 shows the learning curves of the selected acquisition functions. When the entire data pool is queried, the learning curve of any querying method will converge to a single point. Since we are cropping randomly, that point will be slightly above the point on the learning curve of random querying whole images of around $2975/8 \approx 372$ images. In our result, this point is approximately 65%. In the normalized learning curves in figure 2.6, the learning curves of querying image crops performs better because there are less pixels in the image crop data pool. If we allowed multiple crops per image, the maximum mIOU would be closer to the maximum mIOU of whole images query.

Since the number of pixels in the entire data pool is 8 times smaller, the improvement of each acquisition function over the random baseline is diminished. Nevertheless, similar to the results from whole image querying, all aggregation by counting queries outperformed their averaging counterpart. We also selected ODIN with $T = 0.1$, which was one of the best performing T values. It also outperformed max-softmax ($T = 1$) again. This experiment shows that aggregation by counting is a superior aggregation method than averaging regardless of whether the querying unit is a batch of whole images or image crops.

Chapter 3

Discussion and Supplementary Experiments

3.1 Image Classification Experiments

To facilitate rapid experimentation, we simplified our problem to image classification and used a smaller dataset and network architecture. For image classification, we only predict a single label for each image. The dataset we chose was fashion MNIST [45]. It contains a training set of 60000 images and a validation set of 10000 images. Each example is a 28×28 grey-scale image of apparel. Figure 3.1 shows a few example images from the dataset. The number of images is much higher than Cityscapes dataset, be each image is much smaller. There are 10 classes in total, ranging from T-shirts to sneakers. The reason we chose fashion MNIST is that it is one of the standard datasets for benchmarking machine learning algorithms. It is very fast to train, but not overly simple like the digits MNIST [27].

The neural network we used is a simple densely connected two-layer model. The input image is flattened and fed into the first layer, which we can vary the width of. The activation function after the first layer is a linear rectifier [16]. Then, it is passed to the final layer with ten neurons and softmax output. We used Adam optimizer [24], sparse categorical cross entropy as loss function, and percentage accuracy as the performance measure.

The querying methods are always compared between entropy querying and random sampling. We chose entropy querying because it is simple to calculate and has one of the best performances. The image acquisition function for classification is the same as the

Figure 3.1: Example images from fashion MNIST [45]

pixel acquisition function for segmentation. It is much simpler because there is only one output label for each image, so there is no need for aggregation. In each active learning cycle, a query batch size of $n = 500$ images is chosen and added to the training set. For entropy querying, the images with the highest entropy are chosen. We set a fixed number of epochs at 20 as we observed this number to be the amount of training needed for the model to fully converge. The rest of the active learning process is kept mostly the same as image segmentation, with minor modifications to accommodate the new dataset.

3.1.1 Model Capacity

In this experiment, we try to show the effect of model capacity on active learning by varying the width of the first layer of our neural network. The model capacity is a measure of the flexibility of the machine learning model. In neural networks, wider and deeper layers usually have a higher model capacity. We hypothesized that high capacity models would have better performances in active learning.

For the first run, we used 512 neurons in the first layer, which is more than enough to get good performances in our problem. Figure 3.2 plots the learning curves of entropy querying as orange and the random baseline as blue. The x-axis is the number of labeled

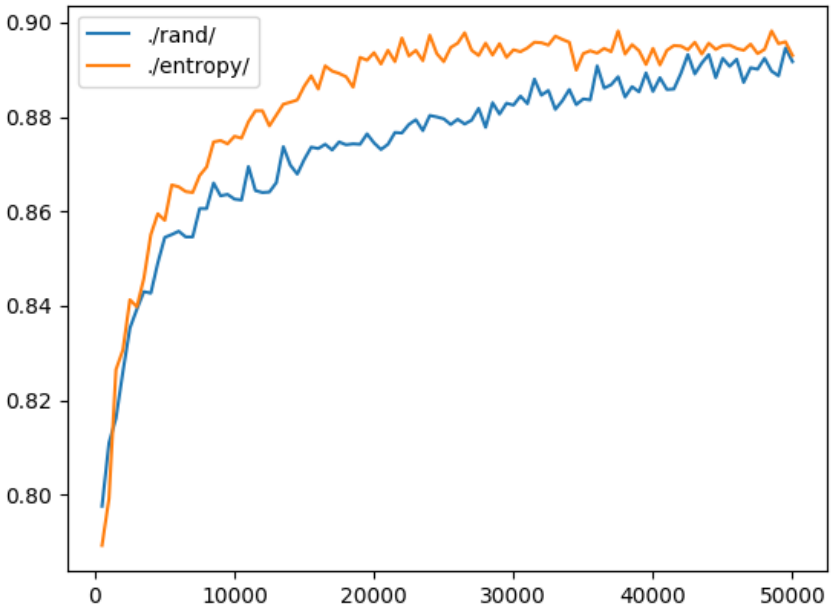


Figure 3.2: Learning curves of entropy querying and the random baseline with 512 neurons in the first layer

images in the training pool, and the y-axis is the percentage validation accuracy. What we see here is the typical behavior of active learning. Entropy querying outperforms the random baseline in almost every cycle.

Next, we repeat the experiment with a varying width of the first layer. Figure 3.3 plots the learning curves of entropy querying and the random baseline with the first layer width 32, 16, 8, and 4. Surprisingly, the results show that starting from a width of 32 neurons, entropy querying starts off performing worse than random sampling in the initial cycles. Then, it catches up to random sampling and eventually surpasses it. We call the point where the learning curve of a querying method surpasses the random baseline the "break-even point".

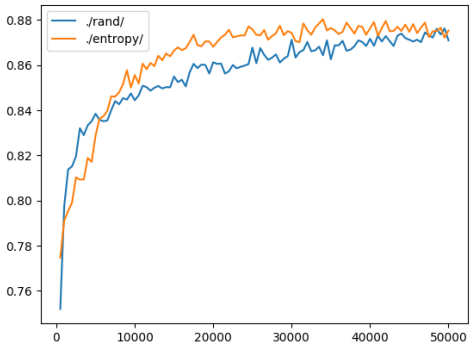
The same pattern shows up again in Figure 3.3b with width of 16 neurons. But this time, the break-even point happens later at around 1000 images rather than 500 images with 32 neurons. The effect of active learning also diminishes. The pattern continues at a width of 8 neurons. Until at 4 neurons, the learning curve of entropy querying never quite surpasses random sampling. The learning curves also get noisier due to underfitting.

These results show that the effect of active learning diminishes as the model capacity decreases, and below some threshold, active learning will not work. This confirms our hypothesis. As the model capacity decreases, the model is less able to learn a complex dataset. Entropy querying picks the images that the current model is uncertain about. When the model's capacity is low, it is better to give it random samples, rather than to challenge it with more difficult samples.

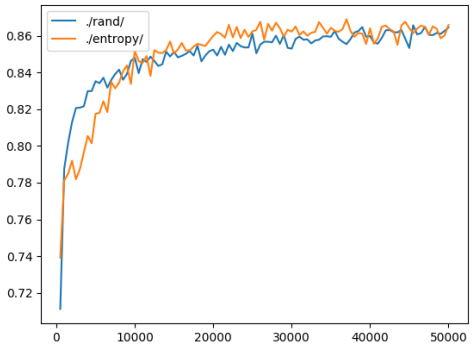
The observation of the break-even effect came as a surprise as we originally thought that active learning should perform better than random sampling in every cycle. Furthermore, the position of the break-even point is also related to the model capacity. As we decrease the model capacity, the break-even point shifts to the right. This seems counterintuitive at first. We think that when the training pool is small, entropy querying fills the training pool with difficult samples. This causes the model unable to perform well. However, these difficult samples pay off in the long run. As the training pool gets larger, the model will eventually gain more confidence on the previously uncertain samples. This only happens when the model's capacity is high. Otherwise, the model never gets the chance to learn the previously uncertain samples well.

3.1.2 Sample Diversity

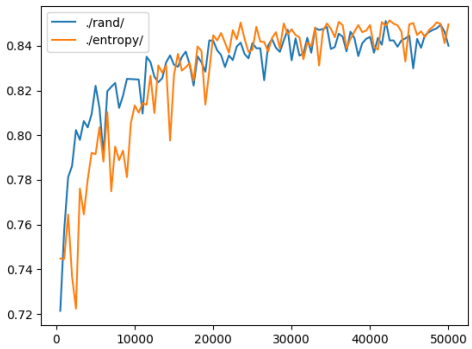
One way to mitigate the problem of low performance of active learning in early cycles is to query images with a mix of high uncertainty and low uncertainty instead of only the



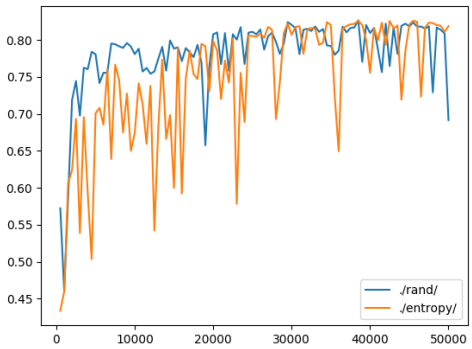
(a) Width = 32 neurons



(b) Width = 16 neurons



(c) Width = 8 neurons



(d) Width = 4 neurons

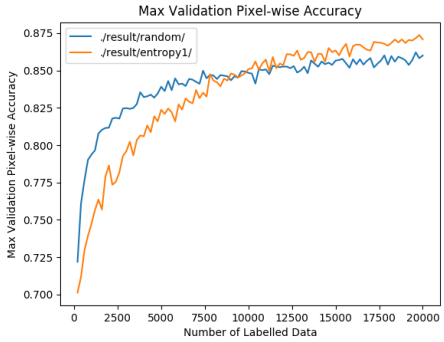
Figure 3.3: Learning curves of entropy querying and the random baseline with varying width in the first layer

highest uncertainty. This is called sample diversity. Y. Fu et al. [13] showed that sample diversity could improve the performance of active learning. We hypothesized that adding sample diversity would improve active learning performances in the early cycles.

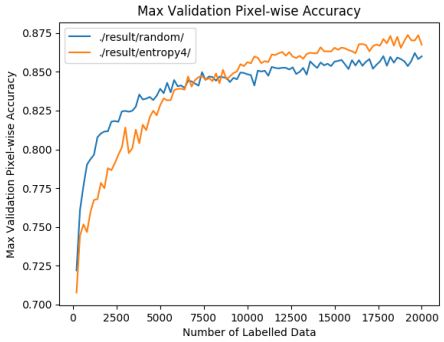
For this experiment, we studied the effect of varying sample diversity. The first layer has a width of 16 neurons so we can see an obvious break-even point. We kept the experiment setup mostly the same as the previous experiment. The only difference is that instead of querying the top $n = 200$ images with the highest entropy, we query every dth highest entropy in strides, meaning skipping every $d - 1$ images in the order from the highest entropy. As an example, for a stride of 4, we would skip every 3 images and query every 4th image. The first image we query is the 4th highest entropy, and the second is the 8th highest entropy. If we get to the image with the lowest entropy, we loop back to the beginning of images that are left.

Figure 3.4 plots the learning curve of entropy querying as orange and random sampling as blue with varying sample diversity. A stride of 1 is the same as the previous experiment with no sample diversity. A stride of d means querying every dth highest entropy. The results show that the break-even point shifts to the left as we increase stride size. This trend continues until we reach a stride of 32, where the break-even point disappears and entropy querying outperforms random sampling in every cycle. As we increase the stride beyond 32 however, the effect of active learning starts to diminish until stride reaches 256, entropy querying has similar performance as random sampling. This makes sense because, at the stride of 256 and a query batch of 200, we are uniformly sampling from the top $256 \times 200 = 51200$ images. This is very close to uniformly sampling from the entire unlabeled pool which is at maximum $60000 - 200 = 58000$ in the second cycle. We are querying just as many as high entropy images as low entropy images. Thus, the performance of entropy querying with a stride of 256 is almost the same as random sampling.

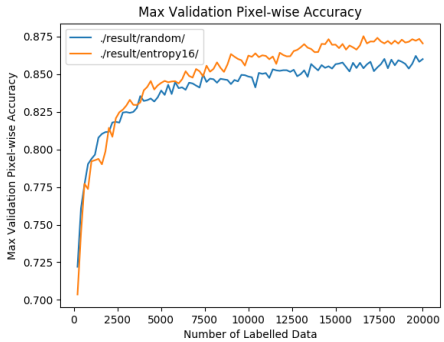
These results show that adding sample diversity does improve active learning performances in the early cycles and shifts the break-even point to the left, which confirms our hypothesis, but only up to a certain threshold. After the threshold, adding diversity would make active learning more and more similar to random sampling. Figure 3.5 plots the break-even point at different sizes of stride. In general, choosing the right amount of sample diversity could be difficult, and we do not have a clear answer to this problem. However, we do know that the optimal diversity depends on the size of the data pool, query batch size, and the labeling budget. If the labeling budget is low, then choose a higher diversity, and vice versa if the budget is high. A good idea is to combine high diversity in early cycles and low diversity in later cycles. Other methods of adding sample diversity are left to explore in future work. If there is a large enough validation set, we can treat sample



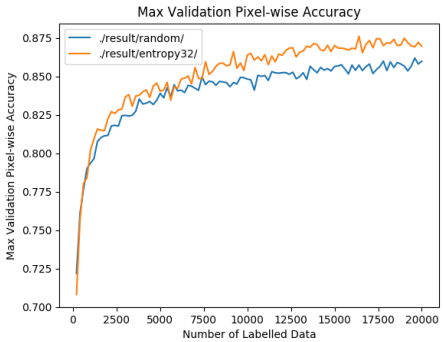
(a) Stride = 1



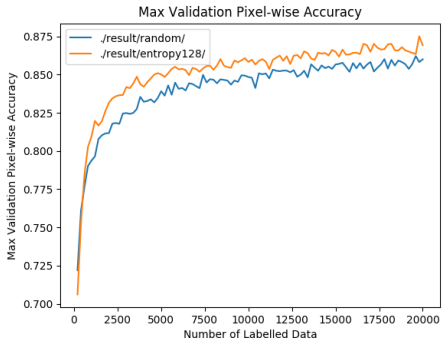
(b) Stride = 4



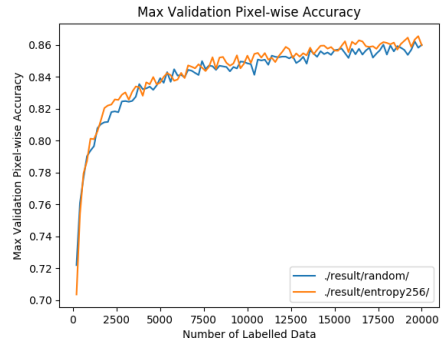
(c) Stride = 16



(d) Stride = 32



(e) Stride = 128



(f) Stride = 256

Figure 3.4: Learning curves of entropy querying and the random baseline with varying sample diversity

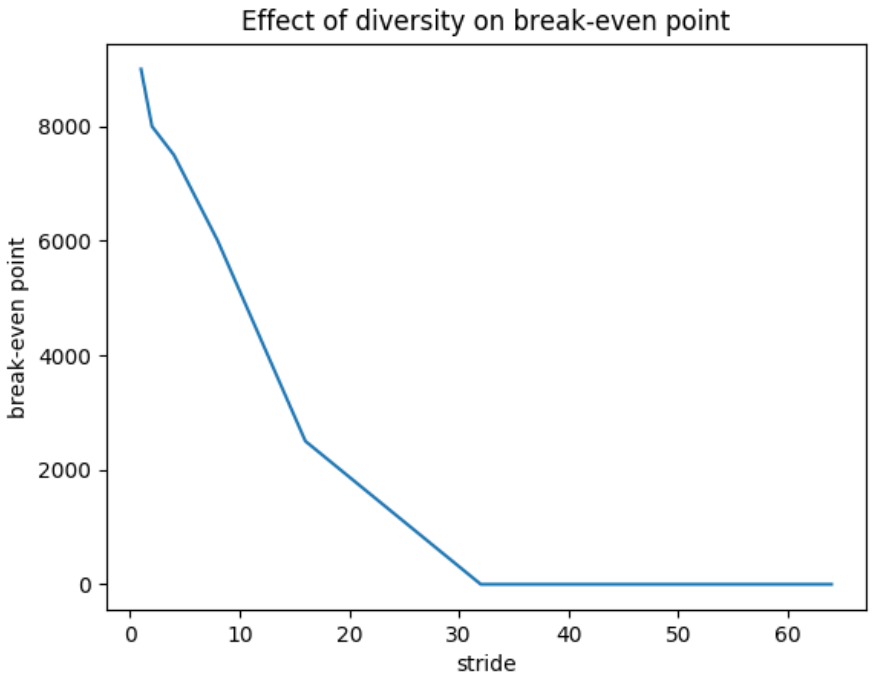
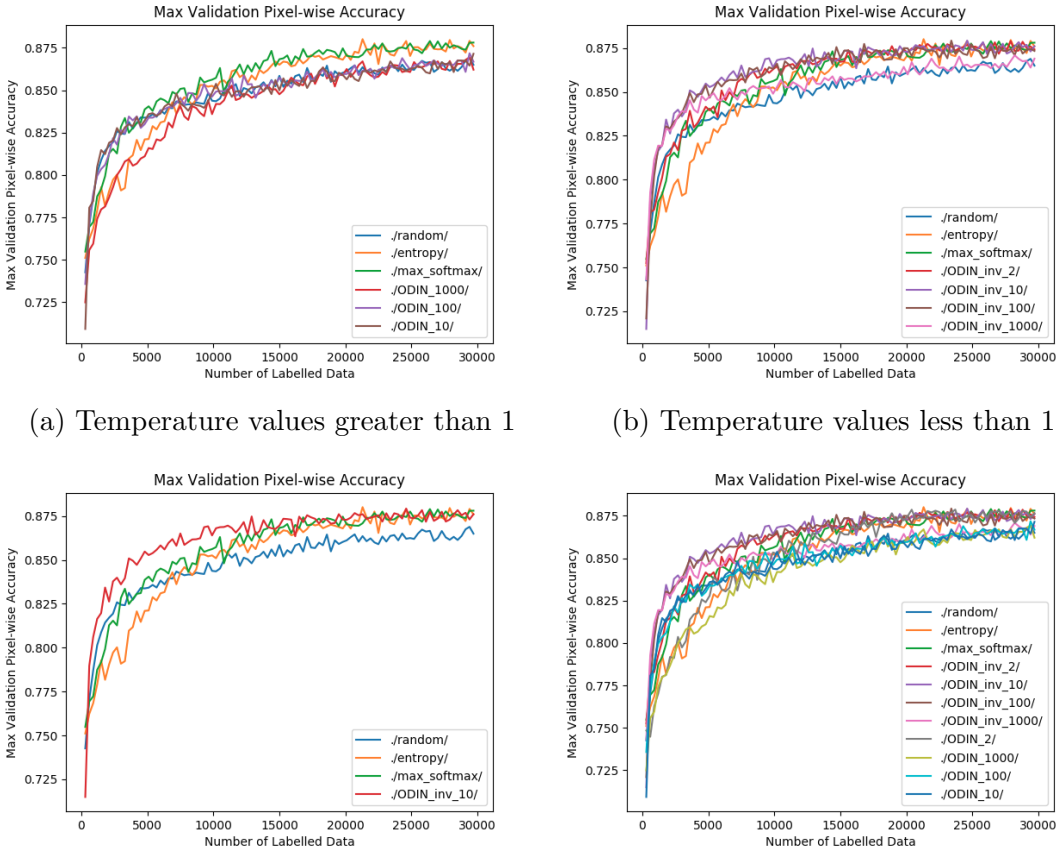


Figure 3.5: Stride size vs. break-even point



(a) Temperature values greater than 1

(b) Temperature values less than 1

(c) Best temperature at $T = 0.1$

(d) Combined temperature values

Figure 3.6: Learning curves of ODIN queries with different temperature

diversity as another hyperparameter to tune.

3.1.3 ODIN Querying

ODIN querying was studied for image segmentation. Because of the faster training on MNIST, we study ODIN querying in more detail for image classification. We are more interested in the effect of temperature T scaling, so we focus on experimenting with different T values and set perturbation *epsilon* to zero in this experiment. The network architecture is the same as the previous experiment with 16 neurons in the first layer. The query batch size is 300 and we run for 100 active learning cycles. Everything else is kept the same as

the previous experiment with no sample diversity.

We experimented with temperature from 0.0001 to 1000 on a logarithmic scale. Figure 3.6 plots the learning curves of ODIN querying with different temperature scaling along with the random baseline and entropy querying as reference. max-softmax querying is equivalent to ODIN querying with $T = 1$. Figure 3.6d plots all the learning curves together. Since it is hard to distinguish the different curves, figure 3.6b and 3.6a plots temperature greater than 1 and less than 1 separately.

A pattern emerges as we take a closer look at the plots. At the higher extreme, $T = 1000$ performs worse than random sampling in the early cycles with less than 1000 images in the labeled pool, and catches up but never really surpasses the random baseline. Decreasing temperature to 100 and 10 improves the performances in the early cycles and makes ODIN querying performs similarly to random sampling. At $T = 1$, or max-softmax querying, the performances of later cycles increased significantly. Compare to entropy querying, max-softmax querying is performing better in the early cycles and similarly in the later cycles. Decreasing the temperature further takes down to T less than 1. At $T = 0.1$ and $T = 0.01$, the performance of ODIN querying in the early cycles improves again. It is better than random sampling for every cycle. The optimal values are 0.1 and 0.01 for temperature scaling. If we decrease the temperature even further, the performance starts to degrade. Interestingly, at $T = 0.001$, the performance of the early cycles stays better than random sampling, but the later cycles perform similar to random.

In summary, a moderate temperature has the best performance in active learning. This is in agreement with the experiments with image segmentation and the artificially generated dataset. The results from Liang et al. [29] where large T having better OOD detection does not apply to active learning. Decreasing the temperature below the optimal shifts the break-even point to the right, where the performance in early cycles degrades faster than the later cycles. Increasing the temperature, however, degrades the later cycles faster than the early cycles. We can imagine this behavior as shifting the break-even point to the left. This behavior is remarkably similar to the sample diversity experiment. The optimal performance for active learning tends to happen when the break-even point has just vanished to the left. This suggests that if the querying method is performing worse than the random baseline in early cycles, tune the hyperparameters such that the break-even point shifts to the left by increase diversity or decrease temperature.

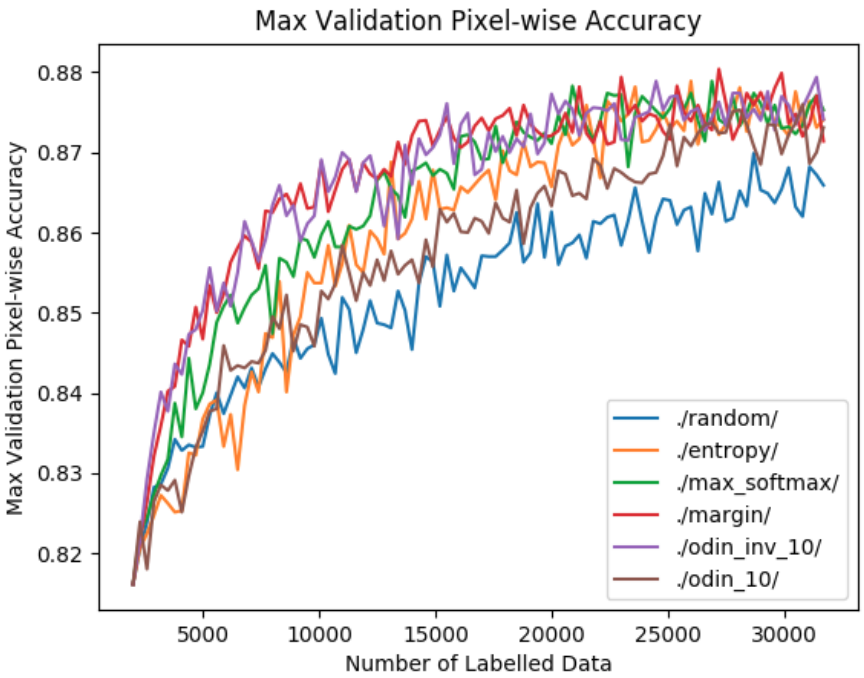


Figure 3.7: Learning curves of different querying methods

3.1.4 Different Querying Methods

To study how well the querying methods used for image segmentation perform on image classification, we further compare the performance of active learning using the querying methods used for image segmentation except for the Bayesian methods. The neural network is the same network with 16 neurons in the first layer. We start with a common initial model trained on 2000 random images. After that, a query batch size of 300 is added to the labeled pool every cycle. The querying methods compared are entropy, max-softmax, margin, ODIN with $T = 0.1$ and $T = 10$.

Figure 3.7 plots the learning curves of these querying methods. From best performance to worst: margin querying and ODIN querying with $T = 0.1$ are the best, then it is max-softmax, ODIN $T = 10$, and the random baseline. The distinction between the learning curves is larger than those for image segmentation. This is likely due to the larger size of the MNIST dataset, and the smaller ratio of query batch size to pool size. Furthermore, entropy querying performs significantly worse than max-softmax in the early cycles, where for image segmentation, entropy and max-softmax are indistinguishable. Nonetheless, the general behavior is quite similar, with all querying methods performing better than random sampling by a significant margin. This confirms that active learning works on both image classification and image segmentation with deep neural networks.

3.2 Optimal Querying

In this section, we try to explore the optimal querying method in the hope of deepening our understanding of active learning. A deterministic querying method can be defined by its querying sequence for a specific dataset. In theory, given any dataset and machine learning model, there must exist at least one subset of the data pool that will give the maximum performance at every labeling budget. However, this does not mean there is a single optimal querying sequence that gives the maximum performance at every labeling budget, since the optimal labeled pool at a later cycle may not be a superset of the labeled pool at an earlier cycle. For example, querying method A can give higher performance than method B in earlier cycles but lower in later cycles. Both A and B can be called optimal if there does not exist a querying method that performs better than A or B in every cycle. Thus, there could be a family of optimal querying methods for a given dataset and model.

Finding an optimal querying sequence by trying every possible sequence is intractable for our datasets. The problem is amplified by the non-deterministic nature of stochastic

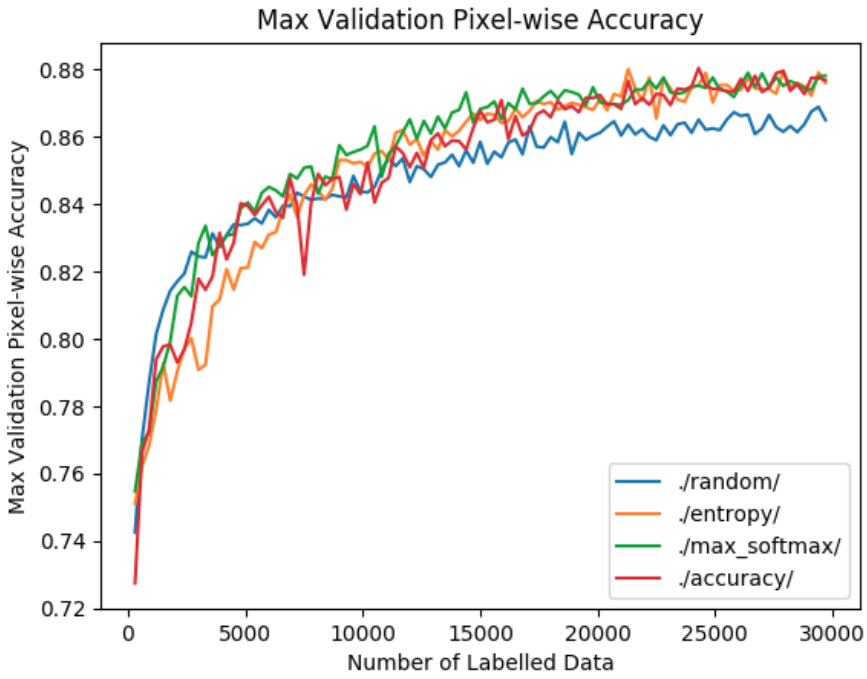


Figure 3.8: Learning curve of querying incorrect samples along with others for reference

gradient descent for optimizing neural networks. In theory, an ideal machine learning model would have a monotonically increasing learning curve, since adding correctly labeled data should not degrade performance. In practice, however, this is often not true.

3.2.1 Querying the Most Incorrect Samples

We thought of ways to get as close to the optimal querying as possible. First, we simplified the problem by allowing the querying method to look at the labels of the unlabeled pool. This is considered "cheating" in the context of active learning since there is no point in active learning if all the data are labeled already, so we are only doing this as an exercise. Note that the machine learning model does not have access to the labels of the unlabeled pool during training. Otherwise, it can get maximum performance in the first cycle.

With the problem redefined, an obvious querying method was to always query the most incorrect samples. We hypothesized was that if the model is always fed with samples that it got wrong, it will keep learning from its mistake. For image classification, there is no

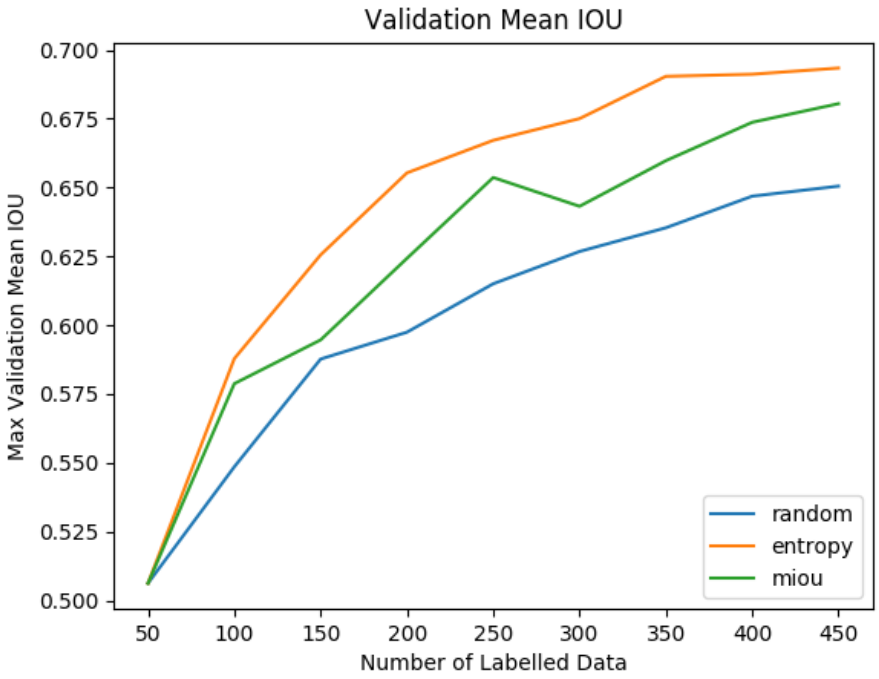


Figure 3.9: Learning curve of querying the lowest mIOU samples along with others for reference

exact measure of the degree of incorrectness. The prediction is either correct or incorrect, so we just randomly query the incorrect samples. If we ever run out of incorrect samples, we would continue sampling random correct samples. Since our simple neural network has a maximum performance of around 90% accuracy, and 10% of the size of dataset 60000 is 6000, any query batch size less than 6000 is most likely to query entirely incorrect samples, especially in the early cycles. The neural network has 16 neurons in the first layer, and the query batch size is 300. Everything else is kept the same as the MNIST experiment. In figure 3.8, learning curve accuracy represents querying incorrect samples. We see that it performed no better than max-softmax querying. Since we know that max-softmax is worse than margin querying, accuracy querying is not the optimal querying method.

In the case of image segmentation, however, we do have an exact measure of the degree of correctness by mIOU. Thus, we performed the same experiment on the Cityscapes dataset with DeepLab by querying images with the lowest mIOU. In figure 3.9, miou represents the learning curve of querying the least mIOU. The results show that querying

the least mIOU performs significantly worse than entropy querying in every active learning cycle.

These results demonstrate that labeling the most incorrect samples is not an optimal querying method, proving our hypothesis incorrect. One plausible reason is that the dataset labels are noisy. If the dataset is not perfectly separable, training on the wrong side of the true decision boundary could lead to lower performance. In the case of fashion MNIST, a bag could be shaped like a skirt, and it would be impossible to tell without more information. Another problem is a truly out-of-distribution sample. These OOD samples are not only out of the distribution of the labeled pool but are also out of the distribution of the entire dataset. In Cityscapes, for example, horse-drawn carriages would appear only a few times. If the horses are labeled as vehicles in the ground truth, training the model to be good at classifying horses adds little value to the overall performance. Another point we need to consider is that machine learning models are not ideal. By that we mean they cannot have perfect accuracy on the test set even when trained on the entire training set, nor could they outperform humans in Cityscapes [7]. (State-of-the-art models do outperform humans in MNIST, making humans not "ideal" in our definition.) This could mean that the model is unable to learn all the information available from the dataset. Even if we provide the model with the most informative sample regarding the true distribution, it may not be the most beneficial to the specific model we are using. All these facts contribute to querying the most incorrect samples not being the optimal querying methods.

3.2.2 Expert Model Query

Another attempt at exploring the optimal querying sequence was to simplify the problem even further. Here, not only do we have access to data labels, but we are also given a trained expert model during querying. This expert is simply the same model trained on the entire dataset. It is the best model we can get with our training pipeline. Note that we are still not allowed to look at the expert model during training. Otherwise, we can get maximum performance by simply copying the expert model's weights. We hypothesized that using the expert model to query would be the optimal query.

The active learning process is mostly the same as the previous MNIST experiments. The only difference is that instead of using the current best model to query the next batch, we use the expert model and query the images with the highest entropy for the expert model. Since we already start with a trained model, the initial training set does not have to be randomly selected. We treat the first active learning cycle as a typical cycle and query as normal.

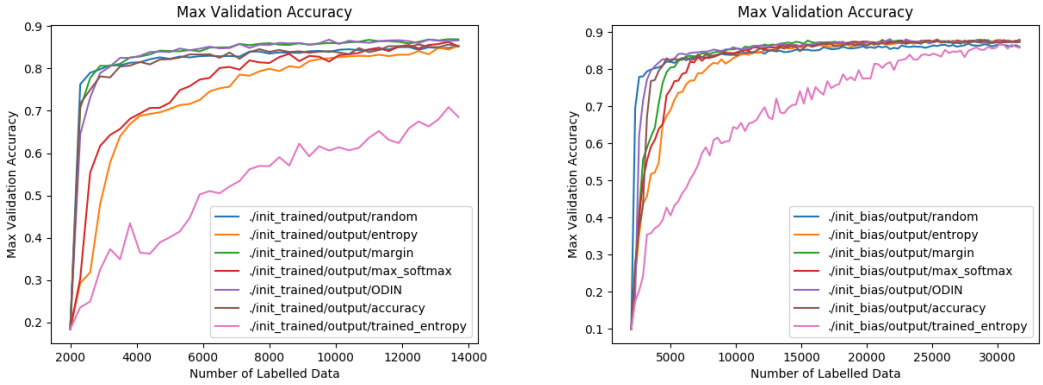


Figure 3.10: Learning curve of querying with a trained expert model along with others for reference

Figure 3.10 plots the learning curve of querying with an expert model by entropy in pink. The right plot is the same as the left plot but with more cycles. Its performance is terrible, and this proves our hypothesis wrong again. The explanation may be similar to the one for querying by mistake. If the expert model is uncertain about the sample, it may not be beneficial to our training.

One interesting side effect from this experiment is that the first initial labeled set gives an extremely low performance of around 20%. This allows us to see how different querying methods "recover" from a bad initial set. Here, random sampling excels. It jumps up to 80% in just a few cycles. This is expected because the next query batch of random sampling is not affected by the previous one. Surprisingly, entropy querying and querying by mistake recover the slowest. Max-softmax is about the same as random sampling, and margin querying and ODIN querying with $T = 0.1$ recovers the fastest.

The right image in figure 3.10 shows the expert model querying does catch up to random eventually. But does it ever surpass the random baseline? To answer this question, we simplified the dataset even further to the digits MNIST. Instead of apparel, digits MNIST is a collection of handwritten digits from 0 to 9. Figure 3.11 shows a few example images from the dataset. All the dataset properties remain the same. We repeated the same experiment on digits MNIST, and plot the results in figure 3.12. We start with an initial labeled set of 300 random images. The learning curve of querying with a trained expert model dips down a little before improving and surpassing the random baseline after 5000 images. This shows that expert model querying is not strictly worse than random sampling.

Another observation is that all the other querying methods outperform random sam-

Figure 3.11: Example images from digits MNIST dataset [27]

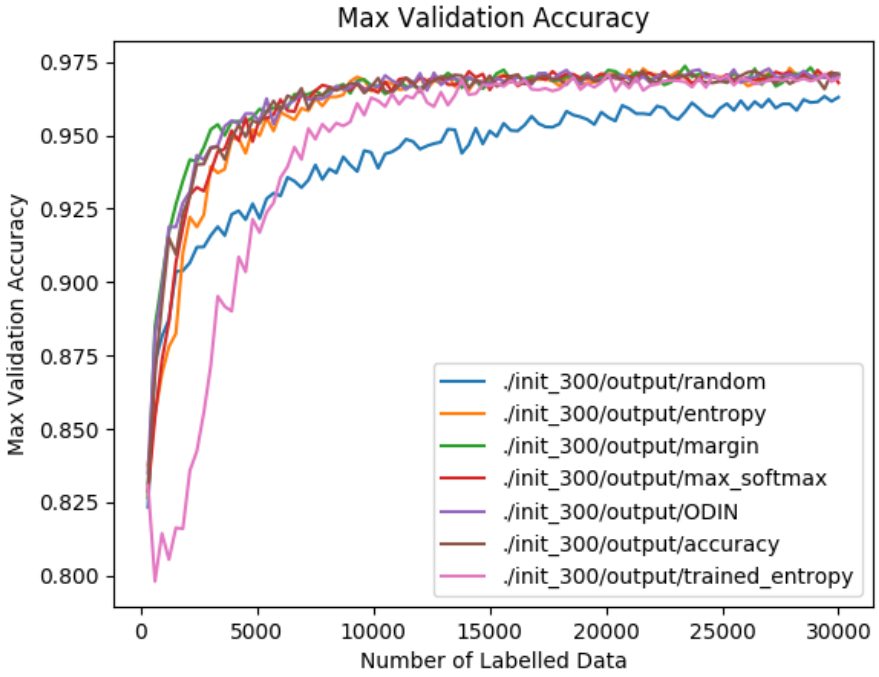


Figure 3.12: Learning curve of querying with a trained expert model along with others for reference on digits MNIST dataset

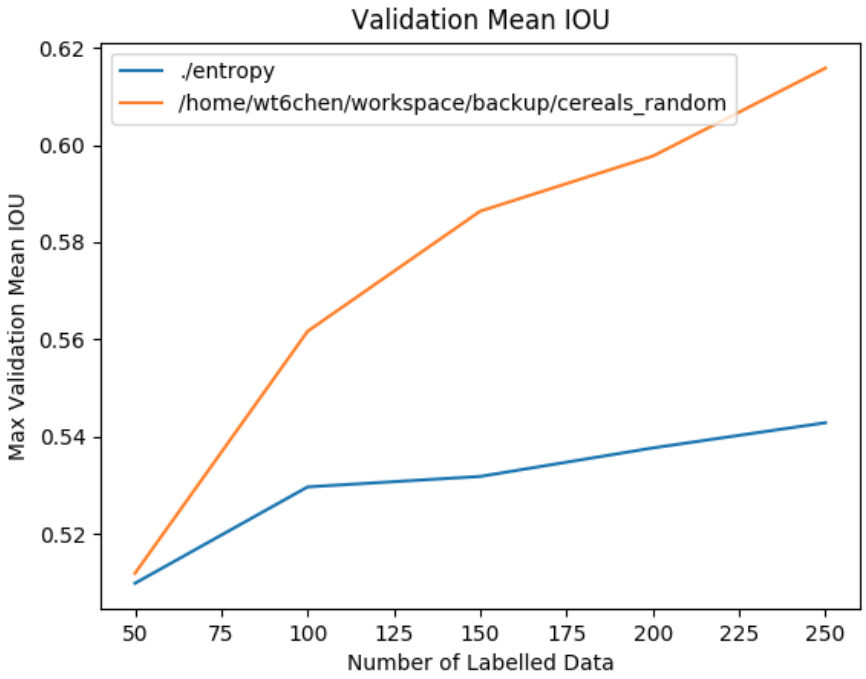


Figure 3.13: Learning curve of inverse entropy querying on Cityscapes

pling even in the early cycles. The break-even point is not apparent. This suggests that lowering the difficulty of the dataset has the effect of shifting the break-even point to the left. This is coherent with the previous observation that increasing the model capacity does the same thing, since lowering the difficulty of the dataset is correlated with increasing the model capacity.

In the case of expert model querying, some querying methods perform much worse than random sampling. This is an important point to emphasize: random sampling is not the worst querying method. In fact, it is quite good among all the possible querying methods.

On the opposite extreme of finding the optimal querying method, finding the worst querying method is just as difficult, if not more difficult. A simple way to get a bad performance is to do the opposite of a querying method with good performance. We take entropy querying as an example and experimented with inverse entropy querying. Instead of querying images with the highest entropy, inverse entropy querying chooses images with the lowest entropy. Figure 3.13 plots the learning curve of inverse entropy querying in blue and the random baseline in yellow for the Cityscapes dataset using DeepLab. Inverse

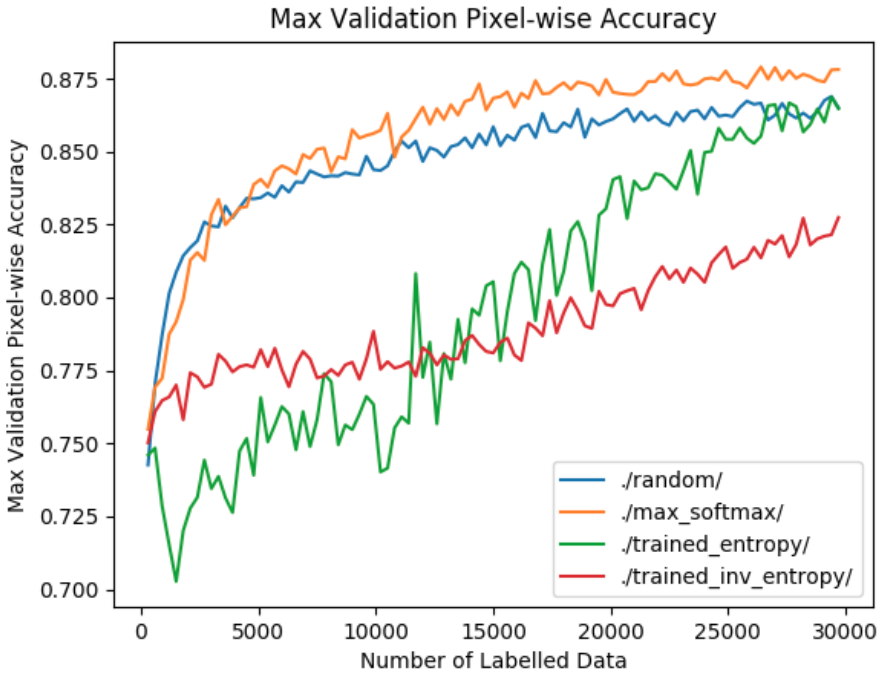


Figure 3.14: Learning curve of inverse entropy querying with trained expert model on fashion MNIST

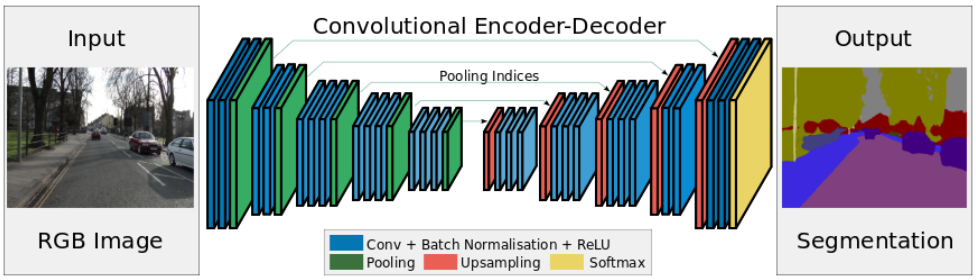


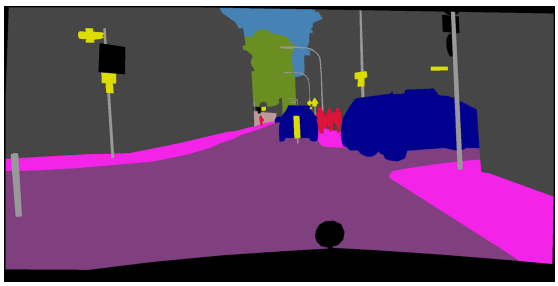
Figure 3.15: Network architecture of SegNet
Source: Adapted from [1]

entropy querying barely improves for 4 cycles and is much worse than random. We further experimented with inverse entropy querying using a trained expert model on fashion MNIST, and plots its learning curve in figure 3.14 along with others for reference. Interestingly, inverse entropy querying using an expert model performs better than entropy expert model querying in the early cycles but gets overtaken in later cycles. These results show that there are querying methods that are much worse than random sampling. However, finding the optimal or the worst querying method with deep learning remains an open problem for future work.

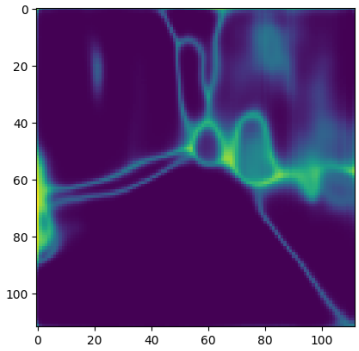
3.3 Experiments with Other Network Architectures

This section illustrates some failure cases for active learning. Most experiments in this section uses SegNet [1] and Fully Convolutional Network (FCN) [31] for image segmentation prediction on Cityscapes dataset. Those experiments were done at the beginning of our research project. We chose these architectures because FCN was used in the thesis by Mackowiak et al. [32], and also because they are relatively fast to train. In the end, we were unable to replicate their results. However, from these experiments, we were able to learn something new, and subsequently devise new experiments that ultimately led to the DeepLab and MNIST experiments. Furthermore, these results also serve as a cautionary tale for future researchers, because they demonstrate the difficulties of active learning on image segmentation. Therefore, although the results were negative, we present them here nonetheless. We also do it in chronological order so the readers can follow the same train of thoughts as we had.

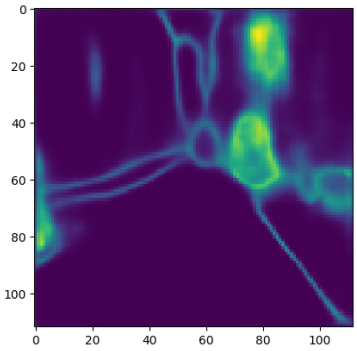
(a) Input image



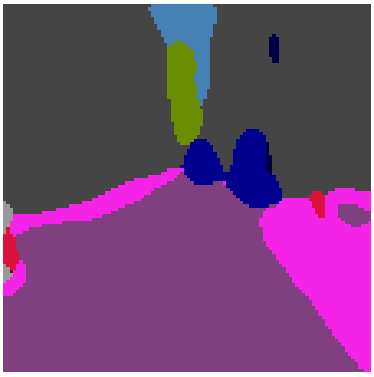
(b) Ground truth



(c) Entropy map

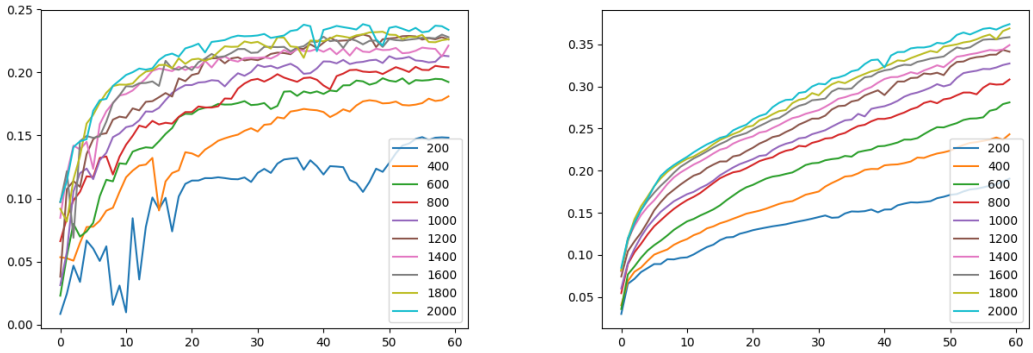


(d) BALD map



(e) Prediction

Figure 3.16: Example output images with SegNet



(a) Validation set (b) Test set

Figure 3.17: Training curves of random sampling in each active learning cycle on validation set and test set

3.3.1 SegNet Architecture Experiment

Training a deep neural network for semantic segmentation is computationally expensive. So the real cost in our research is the training cost rather than labeling cost. We started this project by using SegNet, which is an outdated but fast architecture. SegNet is composed of an encoder part and a decoder part. Each encoder has one or more convolutional layers, batch normalization, ReLU, followed by max-pooling and subsampling [1]. Images are down-sampled in the encoders first then up-sampled in the decoders. Figure 3.15 shows the network architecture of SegNet. To retain the high-frequency detail in prediction, SegNet uses pooling indices in the decoder to perform upsampling. The SegNet we used had a VGG backbone [39] with weights pre-trained on ImageNet [9, 26, 34]. Figure 3.16 shows an example of an input image, ground truth, entropy map, BALD map, and prediction. The input image is scaled down and reshaped into a square before feeding into the network for easier training. The output images, therefore, are also square and downscaled.

The active learning procedure is the same as the DeepLab experiments. Figure 3.17 plots the training curves of random sampling in each active learning cycle on the validation set and the test set. Figure 3.18 plots the learning curves of random sampling, entropy querying, inverse entropy querying, and order querying. Order querying means we query the images in alphabetical order of image names. Since the image names are ordered by the name of the city, order querying always queries in the order of city names. We included this querying method here because it is trivial to implement. The results show that

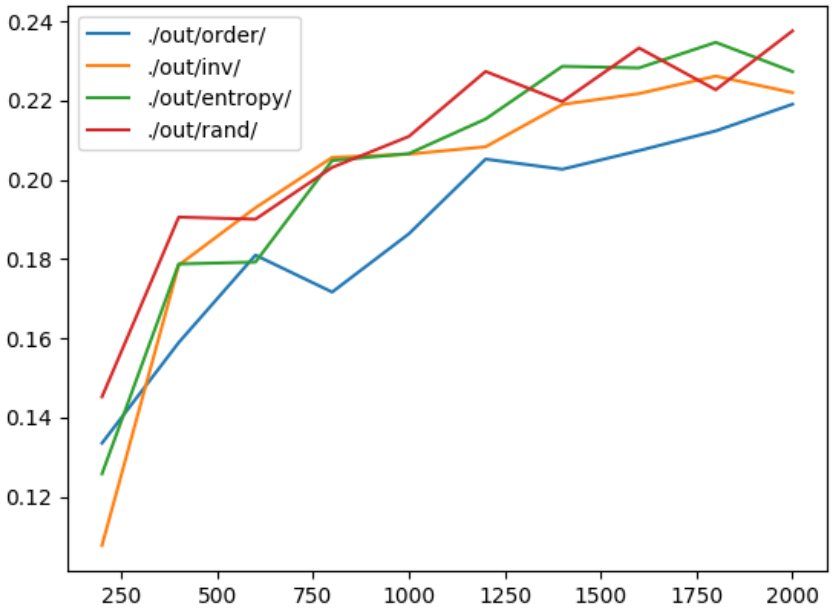


Figure 3.18: Learning curves of random sampling, entropy querying, inverse entropy querying and order querying

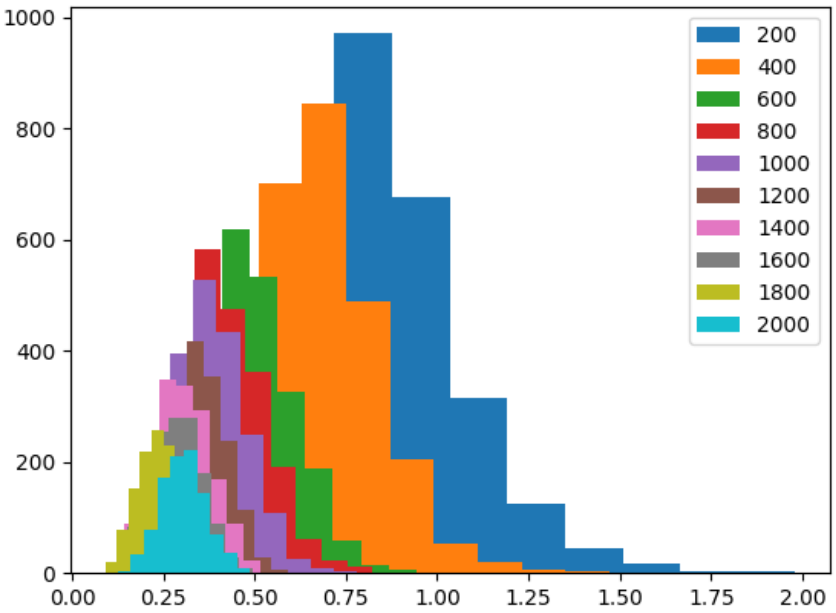


Figure 3.19: Histogram of image entropy acquisition values by averaging of the unlabeled pool in each active learning cycle with the legend showing the number of images in the unlabeled pool

entropy querying does not outperform the random baseline. We also tried inverse entropy querying, which means querying the images with the smallest image entropy instead of the largest. The experimental results show that Inverse entropy querying is about the same as random querying. Interestingly, when the images are queried by city names, it significantly underperformed random sampling. This suggests that images in the same city are more likely to have larger information overlap.

Our first reaction is that there is something wrong with the querying method. One hypothesis was that most images have similar image entropies, but analysis shows that image entropies have a relatively large variance. Figure 3.19 plots the histogram of image entropy in each active learning cycle. The distributions shift to the left and get smaller because the highest entropies are taken and the unlabeled pool left gets smaller in each cycle. Next, we tried BALD. Figure 3.20 shows negative result.

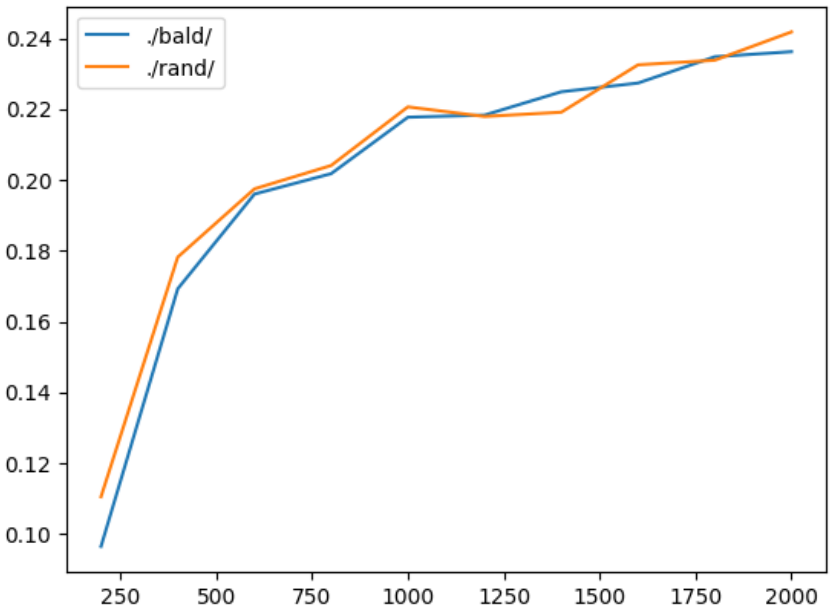


Figure 3.20: Learning curves of BALD querying with the random baseline

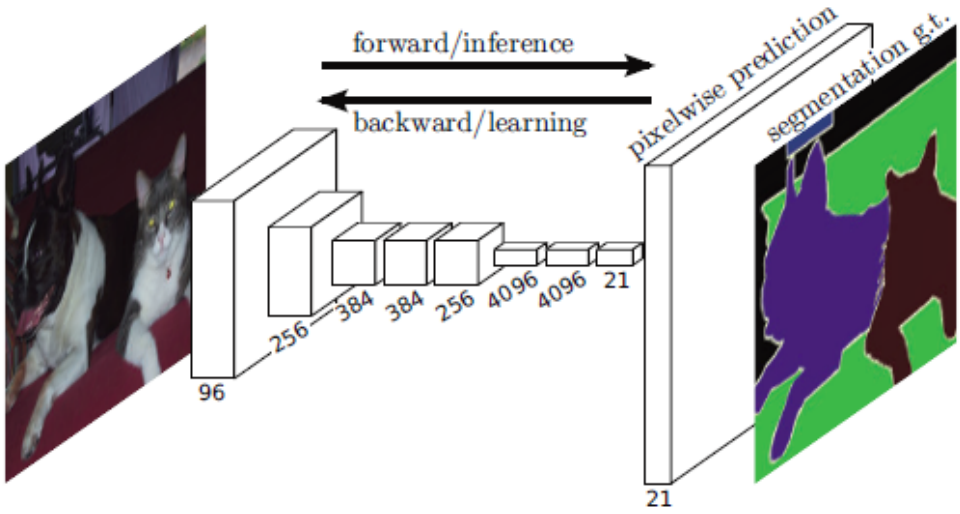


Figure 3.21: Network architecture of FCN
Source: Adapted from [31]

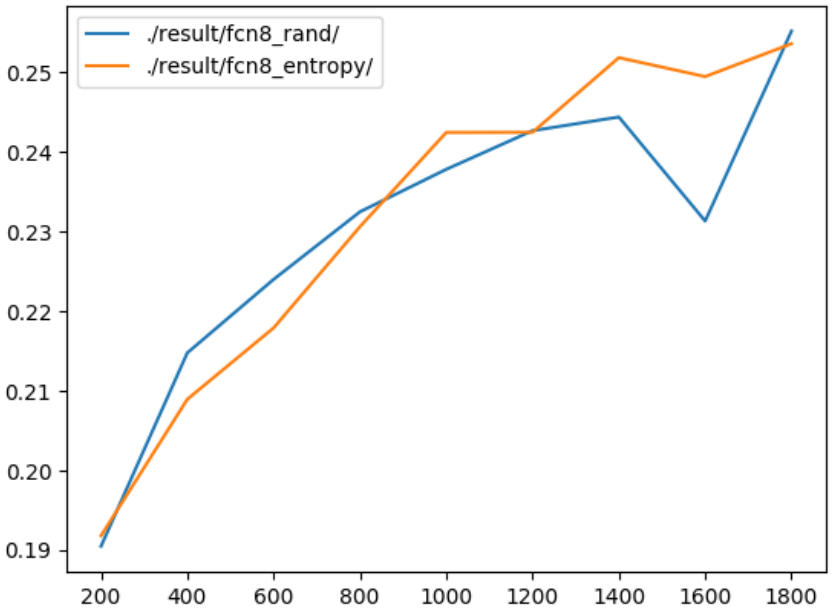


Figure 3.22: Learning curves of entropy querying with the random baseline

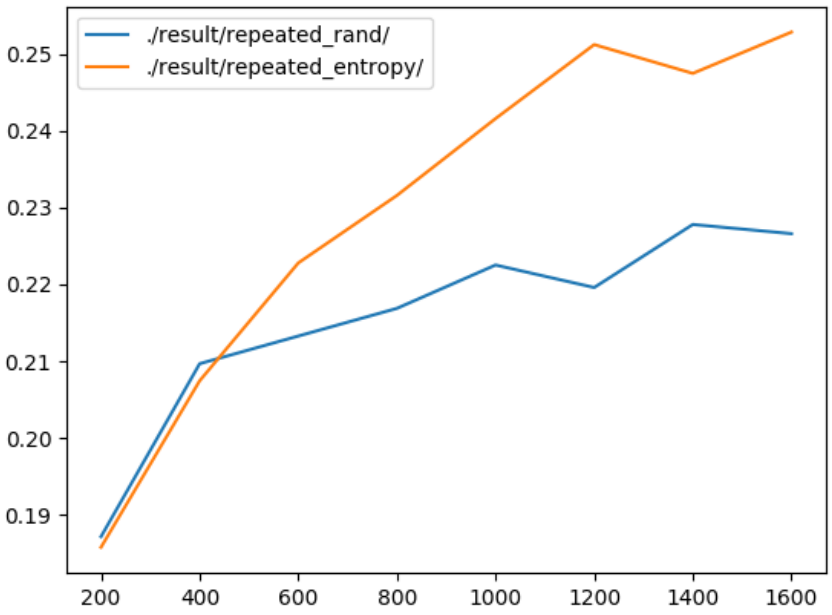


Figure 3.23: Learning curves of entropy querying with the random baseline on repeated dataset

3.3.2 FCN Architecture Experiment

At this point, we started looking for faults other than the querying method. One hypothesis is that the capacity of the neural network is too small. Fully Convolutional Network (FCN) was used to replace segnet (figure 3.22). The experiments were repeated and entropy querying still did not outperform the random baseline (figure 3.22). However, this does not reject the hypothesis as FCN is not a state-of-the-art network.

Another hypothesis was that there are faults with our code. To verify our hypothesis, we simplified the problem to the point where it is trivial. One image from the Cityscapes dataset of 2975 was selected and copied 2975 times. In effect, out of the entire unlabeled pool, about half of them are the same image. Querying that repeated image has zero information gain. Experimental results were positive (figure 3.23), meaning entropy querying did outperform the random baseline. This shows that entropy querying is smart enough to not pick the same image over and over again, and this is strong evidence that suggests

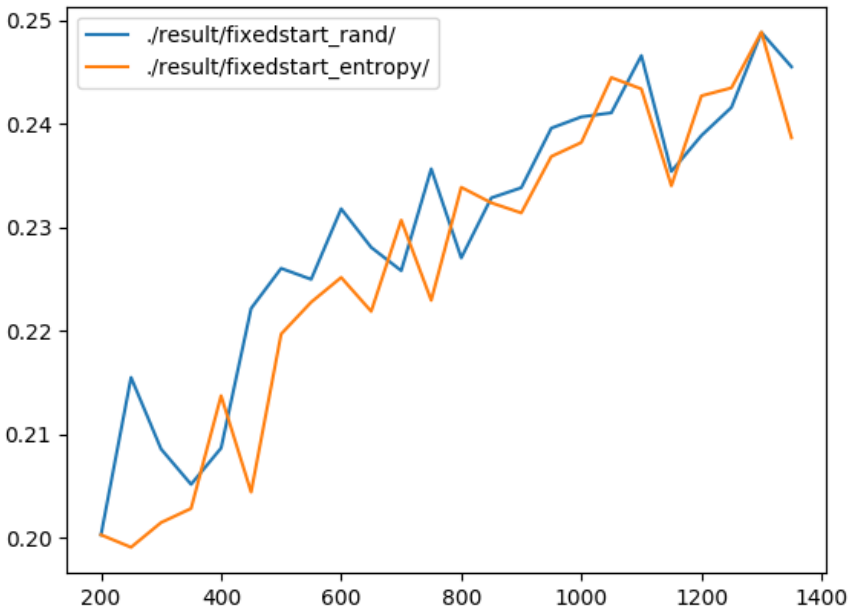


Figure 3.24: Learning curves of entropy querying and random sampling with common initial model

our code is working as intended. Unfortunately, these results are on a biased dataset and not useful in application.

We pointed out two details that could potentially cause errors. The first detail is that the initial network trained from the first query batch using different querying methods are not the same. It was hypothesized that a small perturbation in the initial network can cause a significant difference in later iterations. Therefore, we fixed the initial network and model weights to be the same and branch off from there. The results were negative (3.24). The second detail is that the results are noisy, so a single negative result could be due to pure chance. We ran the experiments 5 times and all of them were negative. With these and all the previous results, we can be confident that this is not a fluke.

It was known to us all along that there is another work, called CEREALS by Mackowiak et al. [32], that claims to have positive results on the same problem. We contacted the authors, but they refused to provide their source code. However, they were able to provide some additional data. Most importantly, they pointed out that they are using a larger

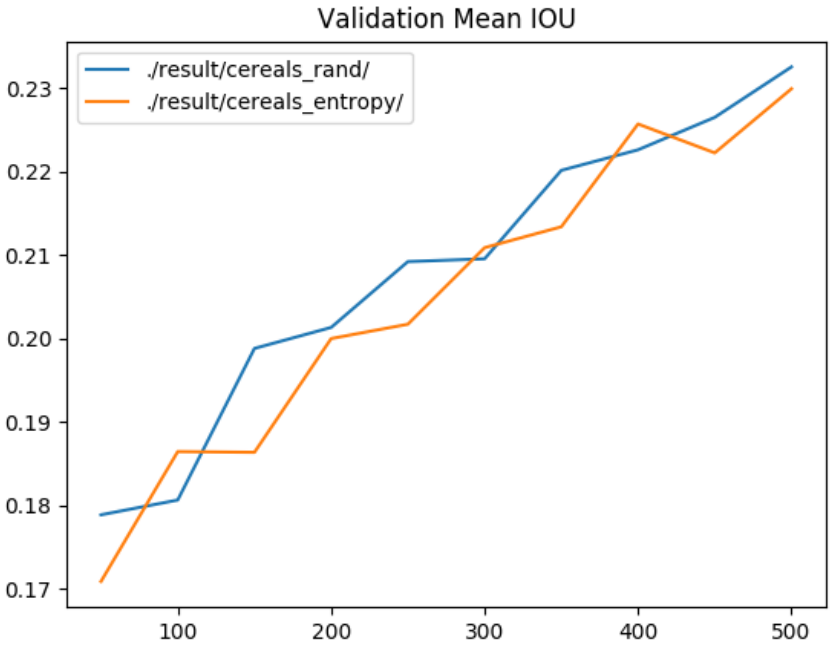


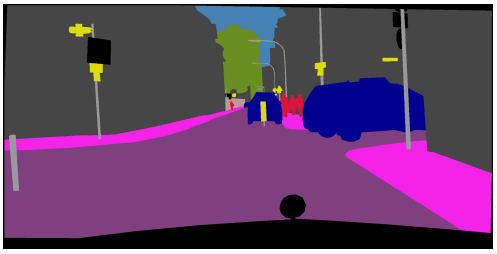
Figure 3.25: Learning curves of entropy querying and random sampling with querying order provided by Mackowiak et al. [32]

version of FCN and a smaller learning rate. Also, they provided us the order in which the images were queried. With these data in hand, we tried to replicate their experiment as closely as possible. We queried the images in the same order and had the same learning rate. However, without the source code, it is very difficult to reproduce the same network. Previously, the images were downsampled 50 times before passing through the network. We tried enlarging the network, but due to limitations in GPU memory, the images were still downsampled 4 times. This experiment is as close a replication as possible with limiting data and computing power. Still, the results were negative (figure 3.25).

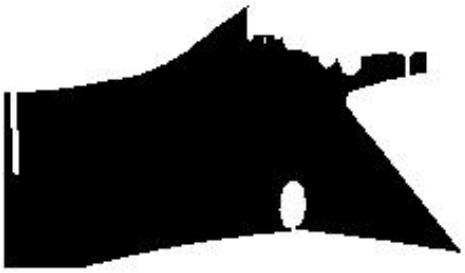
After all the negative results, we decided to do a sanity check on classification problems. This led to the experiments with the MNIST dataset. With the results from the MNIST experiments, the next immediate hypothesis is that the FCN we used is below the critical network capacity because no break-even points were observed. However, due to limitations in computing power, we could not increase the capacity of FCN any further. Instead, we decreased the difficulty of the dataset by reducing the number of label classes to 2 classes: foreground and background. It is important to pick a foreground that exists in most images. Otherwise, the training will be difficult to converge. The Cityscapes dataset originally has 19 different classes. We picked roads and other vehicle surfaces as foreground and everything else as background. Figure 3.26 shows an example input image and its output images. Using FCN, we were able to achieve higher than 90% mIOU using less than 200 training images. However, the results are still negative 3.27.

These failure cases eventually led to success using DeepLab. Although we cannot prove why active learning failed with SegNet and FCN, we strongly believe that it was due to them not having enough model capacity. However, this leaves the question of why the 2 classes case did not work even though the network was able to achieve 90% mIOU, which is even higher than what DeepLab can achieve with all classes. Unfortunately, we are still unable to reproduce the results of Mackowiak et al. exactly. This is worth studying in more detail when they release the source code in the future.

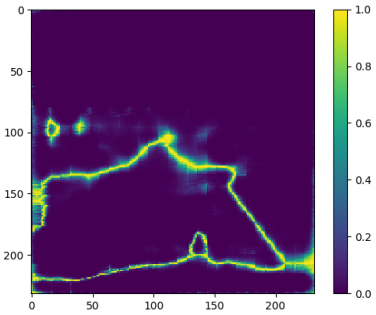
(a) Input image



(b) Ground truth



(c) 2 classes ground truth



(d) 2 classes entropy map

(e) 2 classes prediction

Figure 3.26: Example output images with 2 classes

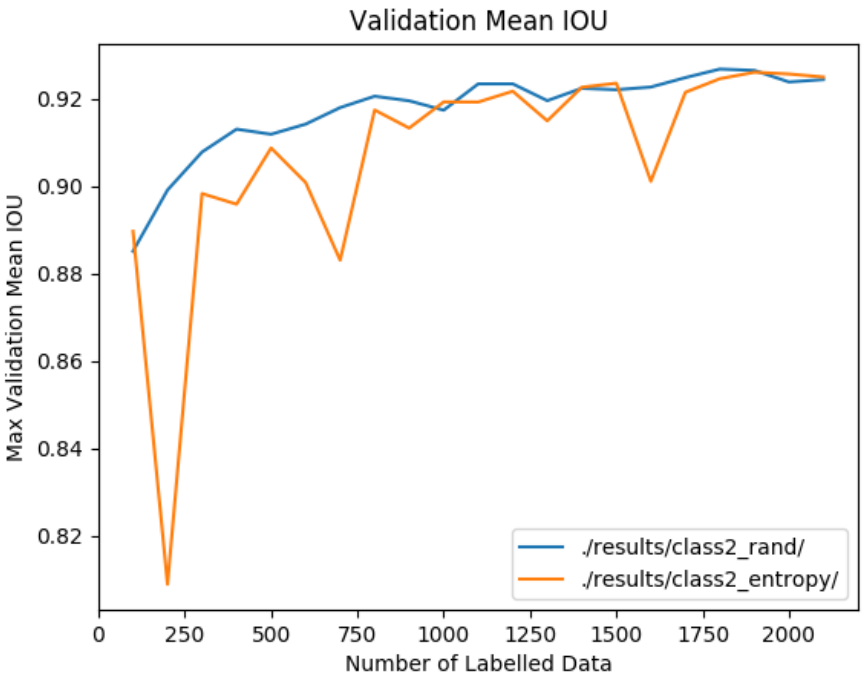


Figure 3.27: Learning curves of entropy querying and random sampling with only 2 classes

Chapter 4

Conclusion and Future Work

4.1 Conclusion

We compared six different querying methods in the context of active learning for image segmentation and found small but discernible differences among them. We demonstrated these results on the industry-standard dataset Cityscapes, as well as randomly generated data, and state-of-the-art image segmentation architecture DeepLab. We also proposed a novel image acquisition value aggregation method by counting with threshold, which we showed to perform better than the standard aggregation by averaging. These findings were repeated with image crop as the querying units, and the results still hold.

Furthermore, we observed an interesting phenomenon where active learning querying methods perform worse than random sampling in the early cycles but overtake random sampling at a break-even point. This break-even point can be shifted to the left by increasing model capacity, adding sample diversity, or tuning temperature scaling for ODIN querying. The performances of the six querying methods also have larger differences than in the case of image segmentation. In the attempt to find the optimal querying method, we found that querying the most incorrect samples and querying with a trained expert model are both suboptimal. Using other network architectures including SegNet and FCN, active learning methods were not able to perform better than random sampling.

4.2 Future Work

One of the major challenges in this research project is of high computational power. Image segmentation with deep learning is already demanding by itself. Active learning on image segmentation increases the computational demand by orders of magnitude. To be able to do something similar in scale to the MNIST experiments for image segmentation would take several months even on high-end GPUs. This makes a study by trial and error infeasible.

Due to the high computational cost, there are several things we were not able to explore. In particular, we leave the evaluation of other image segmentation dataset for future work. The COCO dataset [30], for example, has 200,000 images with 80 classes, which is much larger in scale than Cityscapes. We hypothesize that active learning will perform better with a larger dataset. Regarding datasets, since our research lab focuses on autonomous driving, it would be interesting to collect real video data from a car and perform active learning on the unlabeled data. Although we would need to label the data ourselves, it would be closer to how active learning would be applied in real-life. Since images from video taken from a real car ride would be more correlated to each other, especially for adjacent frames, this is where active learning excels.

Other image aggregation methods with pixels weighted unequally would also be worth exploring. Different approaches to aggregate a single large area of high uncertainty versus multiple small areas could lead to different results. Furthermore, it would be interesting to see the combination of aggregation by counting with other dimensions such as region-based active learning. We believe that such a combination could potentially outperform previous results.

Currently, we do not see any easy way to overcome the computational costs. One potential solution is to retain some information learned from each active learning cycle. However, one must be careful not to "pollute" the knowledge gained in the model from a potentially biased training pool in early active learning cycles. A method that retains information learned but also learns from each training sample equally would be ideal. Thus, until faster training methods, deep active learning will remain a challenging field of study.

References

[1] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

[2] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014.

[3] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

[4] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[5] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018.

[6] David A Cohn, Zoubin Ghahramani, and Michael I Jordan. Active learning with statistical models. *Journal of artificial intelligence research*, 4:129–145, 1996.

[7] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

[8] Ido Dagan and Sean P. Engelson. Committee-based sampling for training probabilistic classifiers. In *In Proceedings of the Twelfth International Conference on Machine Learning*, pages 150–157. Morgan Kaufmann, 1995.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[10] Stefan Depeweg, José Miguel Hernández-Lobato, Finale Doshi-Velez, and Steffen Udluft. Decomposition of uncertainty in bayesian deep learning for efficient and risk-sensitive learning. *arXiv preprint arXiv:1710.07283*, 2017.

[11] Bradley Efron and Robert J Tibshirani. *An introduction to the bootstrap*. CRC press, 1994.

[12] King-Sun Fu and JK Mui. A survey on image segmentation. *Pattern recognition*, 13(1):3–16, 1981.

[13] Yifan Fu, Xingquan Zhu, and Bin Li. A survey on instance selection for active learning. *Knowledge and information systems*, 35(2):249–283, 2013.

[14] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

[15] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1183–1192. JMLR. org, 2017.

[16] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323, 2011.

[17] Marc Gorriz, Axel Carlier, Emmanuel Faure, and Xavier Giro-i Nieto. Cost-effective active learning for melanoma segmentation. *arXiv preprint arXiv:1711.09168*, 2017.

[18] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1321–1330. JMLR. org, 2017.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[20] Robert Hecht-Nielsen. Theory of the backpropagation neural network. In *Neural networks for perception*, pages 65–93. Elsevier, 1992.

[21] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016.

[22] Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*, 2011.

[23] Tejaswi Kasarla, G Nagendar, Guruprasad M Hegde, V Balasubramanian, and CV Jawahar. Region-based active learning for efficient labeling in semantic segmentation. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1109–1117. IEEE, 2019.

[24] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[25] Akshay Krishnamurthy, Alekh Agarwal, Tzu-Kuo Huang, Hal Daumé III, and John Langford. Active learning for cost-sensitive classification. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1915–1924. JMLR. org, 2017.

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[27] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, 2:18, 2010.

[28] David D. Lewis. A sequential algorithm for training text classifiers: Corrigendum and additional data. *SIGIR Forum*, 29(2):13–19, September 1995.

[29] Shiyu Liang, Yixuan Li, and R Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. *arXiv preprint arXiv:1706.02690*, 2017.

[30] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[31] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[32] Radek Mackowiak, Philip Lenz, Omair Ghori, Ferran Diego, Oliver Lange, and Carsten Rother. Cereals-cost-effective region-based active learning for semantic segmentation. *arXiv preprint arXiv:1810.09726*, 2018.

[33] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[34] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[35] Tobias Scheffer, Christian Decomain, and Stefan Wrobel. Active hidden markov models for information extraction. In *International Symposium on Intelligent Data Analysis*, pages 309–318. Springer, 2001.

[36] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*, 2017.

[37] Burr Settles. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2009.

[38] Claude Elwood Shannon. A mathematical theory of communication. *Bell system technical journal*, 27(3):379–423, 1948.

[39] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[40] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[41] David Stutz, Alexander Hermans, and Bastian Leibe. Superpixels: An evaluation of the state-of-the-art. *Computer Vision and Image Understanding*, 166:1–27, 2018.

[42] Simon Tong and Daphne Koller. Active learning for parameter estimation in bayesian networks. In *Advances in neural information processing systems*, pages 647–653, 2001.

[43] Sudheendra Vijayanarasimhan and Kristen Grauman. What's it going to cost you?: Predicting effort vs. informativeness for multi-label image annotations. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2262–2269. IEEE, 2009.

[44] Keze Wang, Dongyu Zhang, Ya Li, Ruimao Zhang, and Liang Lin. Cost-effective active learning for deep image classification. *IEEE Transactions on Circuits and Systems for Video Technology*, 27(12):2591–2600, 2016.

[45] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[46] Lin Yang, Yizhe Zhang, Jianxu Chen, Siyuan Zhang, and Danny Z Chen. Suggestive annotation: A deep active learning framework for biomedical image segmentation. In *International conference on medical image computing and computer-assisted intervention*, pages 399–407. Springer, 2017.